BASIC APPROFONDI

ALGORITHMES MATHÉMATIQUES ET PHYSIQUES

P. BEAUFILS / W. LUTHER

SYBEX

BASIC APPROFONDI
ALGORITHMES, MATHÉMATIQUES, PHYSIQUE

POUR UN CATALOGUE COMPLET DE NOS PUBLICATIONS

FRANCE

6-8, Impasse du Curé
75881 PARIS CEDEX 18
Tél. : (1) 42.03.95.95
Télex : 211801

U.S.A.

2021 Challenger Drive, 100
Alameda, CA 94501
Tél. : (415) 523-8233
Télex : 336311

ALLEMAGNE

Vogelsanger. Weg 111
4000 Düsseldorf 30
Tel. : (211) 61.80.2-0
Telex : 8588163

P. BEAUFILS / W. LUTHER

BASIC APPROFONDI

ALGORITHMES, MATHÉMATIQUES, PHYSIQUE

SYBEX

Paris • San Francisco • Düsseldorf • Londres

ISBN 2-7361-0149-9

SOMMAIRE

IV
INTERPOLATION

V
EQUATIONS DIFFERENTIELLES

VI
UTILITAIRES

INTRODUCTION

Cet ouvrage propose une collection d'algorithmes numériques, destinés à être employés sur micro-ordinateur dans le but de résoudre nombre de problèmes scientifiques, relatifs aux mathématiques, à la physique et à l'électronique, et s'adresse à tous les possesseurs de micro-ordinateurs, quel que soit leur type. Tous ces programmes ont d'ailleurs été mis au point sur une version 16 K. Ils sont écrits en BASIC standard et ne font pas appel aux propriétés particulières d'une machine.

Ce livre s'adresse à tous les curieux, bloqués devant un problème qu'ils ne savent pas résoudre. C'est alors que l'on s'aperçoit que l'explosion de l'informatique, et les nouveaux modes de pensée qu'elle devrait induire, n'ont pas toujours été accompagnés par la réforme nécessaire de l'enseignement théorique dans nos lycées et facultés.

Prenons l'exemple (ô combien classique!) du pendule pesant, au programme des classes de terminale. Les connaissances des élèves leur permettent aisément d'écrire l'équation différentielle du mouvement; il s'agit en effet d'un problème simple de mécanique.

Un double écueil surgit alors : d'une part, cette équation n'est pas linéaire — on décide alors, assez arbitrairement, de n'étudier que les « petits » mouvements —; d'autre part, les connaissances mathématiques des élèves permettant de résoudre cette nouvelle équation linéarisée ne sont pas du niveau de cette classe; les mathématiciens fournissent alors la solution (signalons à leur décharge qu'ils peuvent mener une discussion qualitative de l'équation exacte et décrire les conditions initiales conduisant à des solutions périodiques).

Or, grâce à l'informatique, la seule connaissance de la première équation, établie par les élèves et comprise par eux, permet d'aboutir, sans aucune approximation, à la solution (numérique), seul objet digne d'intérêt pour le physicien.

Qu'importe la méthode! Et n'est-il pas merveilleux de trouver en quelques secondes la solution de problèmes que des dizaines de générations ont étudiés péniblement?

Nous nous sommes particulièrement attachés à proposer des programmes courts et modulaires, sans pour autant sacrifier la précision ou la

rigueur des algorithmes; nous expliquons toujours largement la méthode employée. En outre, quand cela était possible, nous avons éliminé les recherches de convergence (la solution existe-t-elle?), puisque tout problème physique a une solution. Tous les programmes ont naturellement été testés, notamment sur le choix d'exemples proposés, et sont utilisables directement pour d'autres applications. Aucune digression théorique n'est faite, surtout si elle ne débouche que sur une méthode inexploitable. Un dernier problème, pratique, subsiste : toutes les méthodes proposées mettent d'autant plus de temps à s'effectuer qu'une grande précision leur est demandée. Il est alors conseillé de travailler sur un micro-ordinateur disposant d'un BASIC compilé, ou, si ce n'est pas possible, d'un Pascal. Nous donnons d'ailleurs les programmes les plus « lents » dans ces deux langages.

Enfin, tous ces programmes sont interactifs et se complètent. Nous donnons à titre d'exemple la combinaison des Chapitres 5 et 6, qui permet d'obtenir d'une façon très élégante la solution d'un problème particulier.

Nous remercions nos élèves respectifs de nous avoir incités à écrire ce document et de nous avoir guidés dans sa réalisation.

PIERRE-MARC BEAUFILS,
WOLFRAM LUTHER.

R E M A R Q U E

Les instructions relatives au mode *texte* sont standard. Pour le mode *haute résolution*, les caractéristiques sont les suivantes (elles demandent donc éventuellement une adaptation) :

- Axe OX horizontal gradué de 0 à 239.
- Axe OY vertical gradué de 0 à 199.
- Origine des axes située en haut à gauche.
- CURSET X, Y, 1 : allume un pixel de coordonnées absolues X et Y.
- CURMOV X, Y, 1 : déplace le point courant des quantités X et Y relativement à l'ancienne position.
- DRAW X, Y, 1 : trace une droite de coordonnées X et Y à partir du dernier point.
- CHAR A, 0, 1 : écrit le caractère de code ASCII A à la position du point courant.
- CIRCLE R, 1 : trace un cercle de rayon R ayant pour centre le dernier pixel allumé.

I

UN ALGORITHME? MAIS C'EST TRÈS SIMPLE!

1. Nombres complexes. Module et phase
2. Équation différentielle linéaire du premier ordre
3. Mouvement ponctuel newtonien
4. Régression linéaire

1. NOMBRES COMPLEXES. MODULE ET PHASE

En électronique, il est souvent fait usage de la notation complexe pour représenter une grandeur relative à un fonctionnement d'un système en régime permanent sinusoïdal. En particulier, la transmittance d'un quadripôle, rapport entre tension de sortie et tension d'entrée, est une grandeur complexe, fonction d'un paramètre, la pulsation du signal sinusoïdal incident. De façon générale, et dans les cas les plus simples, cette transmittance peut être de trois types fondamentaux :

1. passe bas du second ordre (1),
2. passe haut du second ordre (2),
3. passe bande du second ordre (3).

Leur expression mathématique est la suivante :

$$\overline{T_1} = \frac{1}{1 + 2\,jm\,\dfrac{\omega}{\omega_0} + \left(j\,\dfrac{\omega}{\omega_0}\right)^2}$$

$$\overline{T_2} = \frac{\left(j\,\dfrac{\omega}{\omega_0}\right)^2}{1 + 2\,jm\,\dfrac{\omega}{\omega_0} + \left(j\,\dfrac{\omega}{\omega_0}\right)^2}$$

$$\overline{T_3} = \frac{2\,jm\,\dfrac{\omega}{\omega_0}}{1 + 2\,jm\,\dfrac{\omega}{\omega_0} + \left(j\,\dfrac{\omega}{\omega_0}\right)^2} = \frac{1}{1 + \dfrac{j}{2\,m}\left(\dfrac{\omega}{\omega_0} - \dfrac{\omega_0}{\omega}\right)}$$

Les électroniciens étudient ces fonctions dans le plan de Bode, plan dans lequel l'axe des abscisses représente le logarithme de la pulsation et celui des ordonnées $20 \log |\overline{T}|$ (en dB donc).

Tracer ces fonctions ne présente pas de difficultés particulières. Cependant, l'étude de la phase pose un problème dans la mesure où le dénominateur commun à ces transmittances a un argument variant de 0 à $-\pi$ lorsque ω varie de 0 à l'infini. Pour la pulsation ω_0, on constate en effet que la tangente de cet argument devient infinie. Il est alors préférable de se servir de la seconde forme de l'expression de $\overline{T_3}$ qui, elle, ne possède pas de singularité gênante, et d'écrire :

$$\operatorname{Arg}(\overline{T_1}) = -\frac{\pi}{2} + \operatorname{Arg}(\overline{T_3})$$

$$\operatorname{Arg}(\overline{T_2}) = \frac{\pi}{2} + \operatorname{Arg}(\overline{T_3})$$

Nous proposons de tracer les lieux de Bode de ces trois types de filtre. Il est possible d'en déduire la réalisation de courbes universelles.

La construction de l'échelle logarithmique est effectuée de la manière suivante :

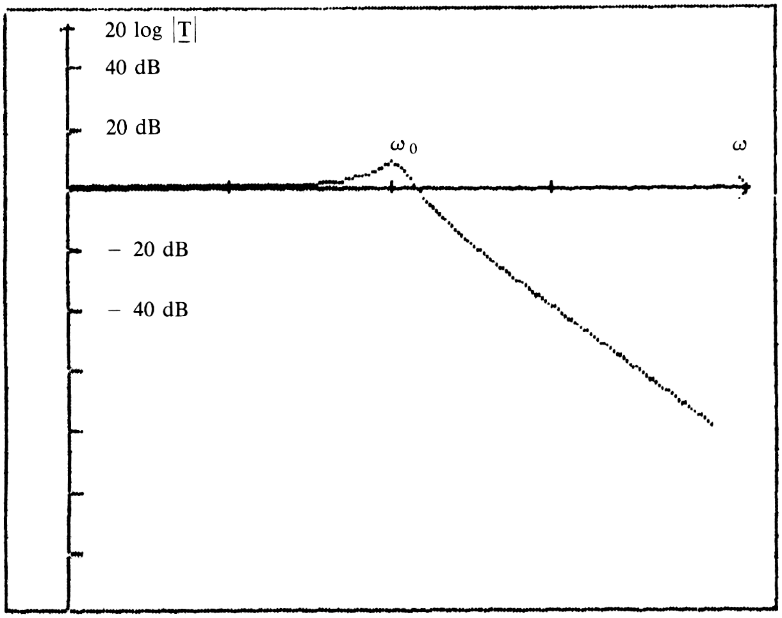
Supposons que 50 points représentent une décade. Nous définissons un incrément $D = 10 \uparrow \left(\frac{1}{50}\right) = 10 \uparrow 0{,}02$. Nous passons donc d'un point au suivant en multipliant la fréquence par cet incrément. Il paraît ainsi évident qu'au bout de 50 points : $f' = f \cdot (D) \uparrow 50 = 10 f$. On peut donc dire que la distance séparant deux points représente $\frac{1}{50}$ de décade.

Nous proposons dans le programme deux graduations possibles pour cet axe horizontal, soit 100 points = 1 décade (on observe ainsi l'évolution des courbes sur deux décades), soit 50 points/décade (observation de quatre décades).

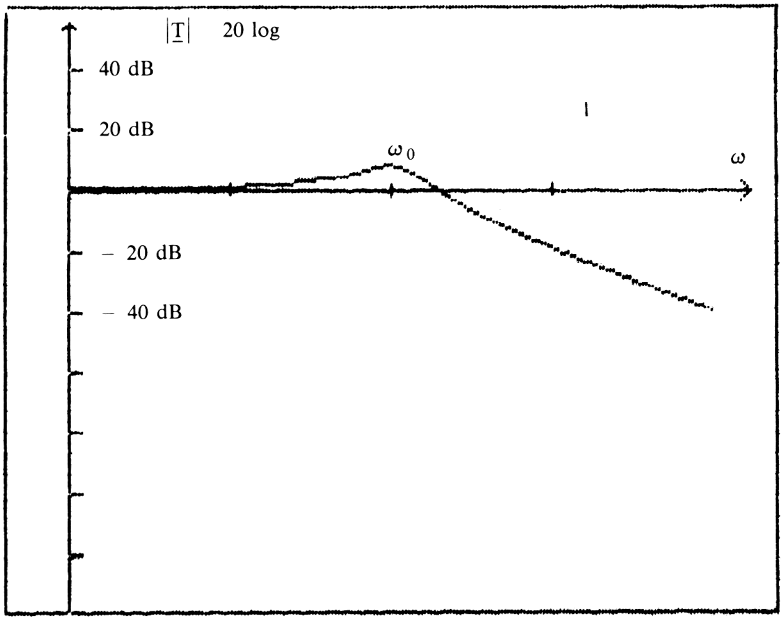
```
 10 CLS
 20 PRINT"TYPE DU FILTRE :"
 30 PRINT
 40 PRINT"        *Passe bas......(1)"
 50 PRINT"        *Passe haut.....(2)"
 60 PRINT"        *Passe bande....(3)"
 65 GET R
 70 PRINT
 80 INPUT"Facteur d'amortissement:";M
 90 IF M<1THENTR=20*LOG(2*M/SQR(1-M^2))
ELSE TR=0
 92 PRINT:PRINT"Voulez vous etudier la f
onction sur 4 decades(1) ou 2 decades(2)
?":GETN
 100 IF N=1 THEN D=10^0.02 ELSE D=10^0.0
1
 110 IF N=1 THEN W=1/(100*D) ELSE W=1/(1
0*D)
 120 ON R GOSUB 300,320,340
 130 DEF FNP(X)=-ATN((X-1/X)/(2*M))+P
 140 PRINT:PRINT"Voulez vous le module(M
) ou la Phase(P)?":GET A$:HIRES
 145 GOSUB 1000
 150 IF A$="M"THEN 160 ELSE IF A$="P" TH
EN 200
 155 GOTO 140
 160 FOR X=0 TO 199
 170 CURSET20+X,60-20*LOG(FNM(W)),1
 180 W=W*D
 190 NEXT
 195 STOP
 200 FOR X=0 TO 199
 210 CURSET20+X,60-(30/(PI/2))*FNP(W),1

 220 W=W*D
 230 NEXT
 240 STOP
 300 DEFFNM(X)=1/(((1-X*X)^2+(2*M*X)^2)^
0.5):P=-PI/2
 310 RETURN
 320 DEF FNM(X)=X^2/(((1-X*X)^2+(2*M*X)^
2)^0.5):P=PI/2
```

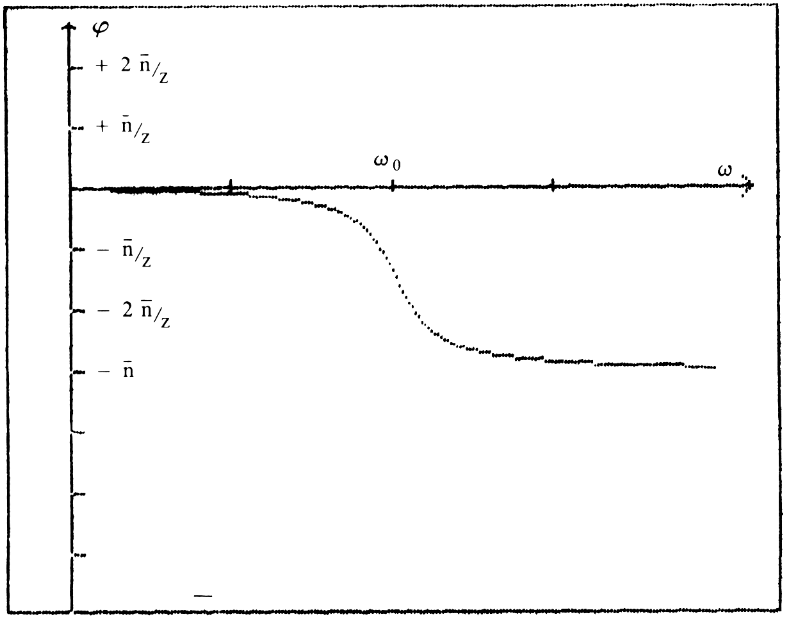
```
 330 RETURN
 340 DEF FNM(X)=2*M*X/(((1-X*X)^2+(2*M*X
)^2)^0.5):P=0
 350 RETURN
 1000 REM *** AXES ***
 1010 CURSET0,0,1:DRAW239,0,1:DRAW0,199,
1:DRAW-239,0,1:DRAW0,-199,1
 1020 CURSET20,60,0:DRAW210,0,1:DRAW-3,-
3,0:CHAR62,0,1
 1030 CURSET20,199,0:DRAW0,-190,1:DRAW-2
,-4,0:CHAR94,0,1
 1040 FOR N = 1 TO 3
 1050 CURSET 20+50*N,62,0:DRAW0,-4,1
 1060 NEXT
 1070 FOR N = 0 TO 8
 1080 CURSET20,20+20*N,0:DRAW 4,0,1
 1090 NEXT
 1100 RETURN
```



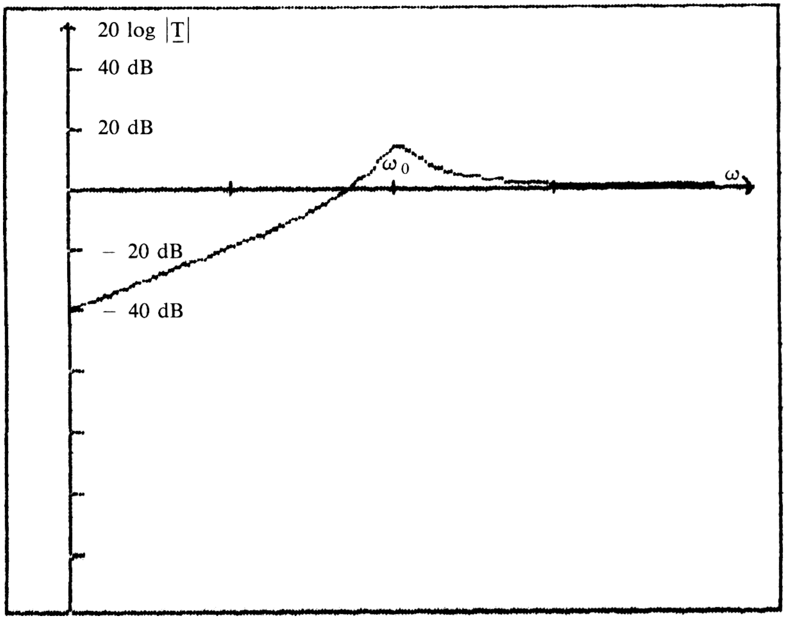
Filtre passe bas. Courbe de module. m = 0,2. *Étude sur 4 décades.* $T_R = 8{,}13$ dB.



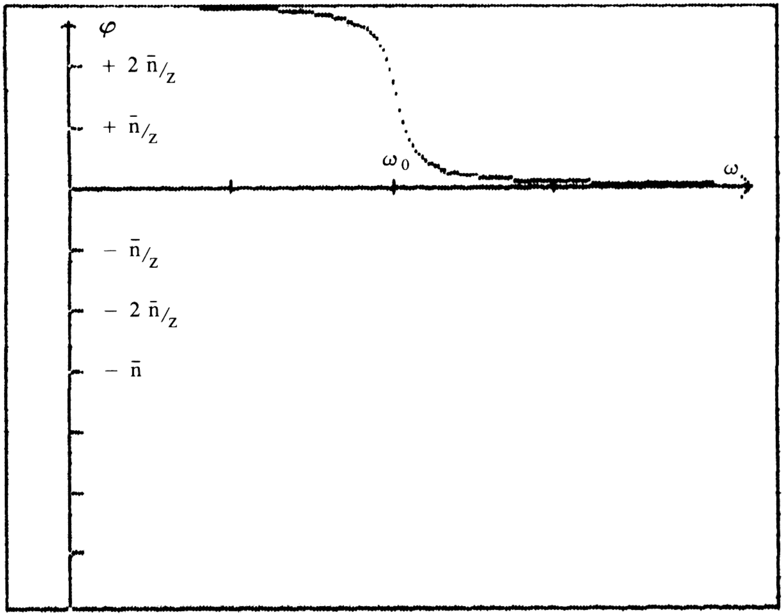
Filtre passe bas. Courbe de module 2 décades. m = 0,2. $T_R = 8{,}13$ dB.



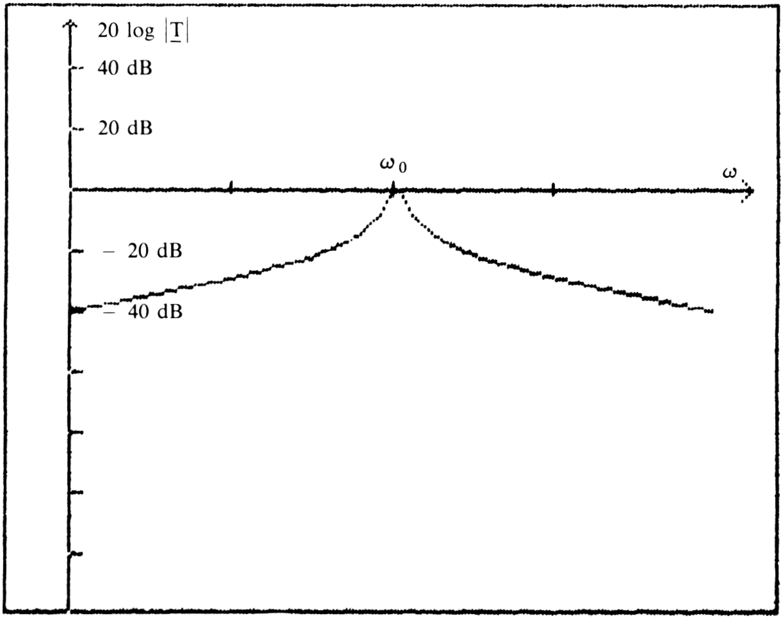
Filtre passe bas. m = 0,2. *Courbe de phase. Étude sur 2 décades.*



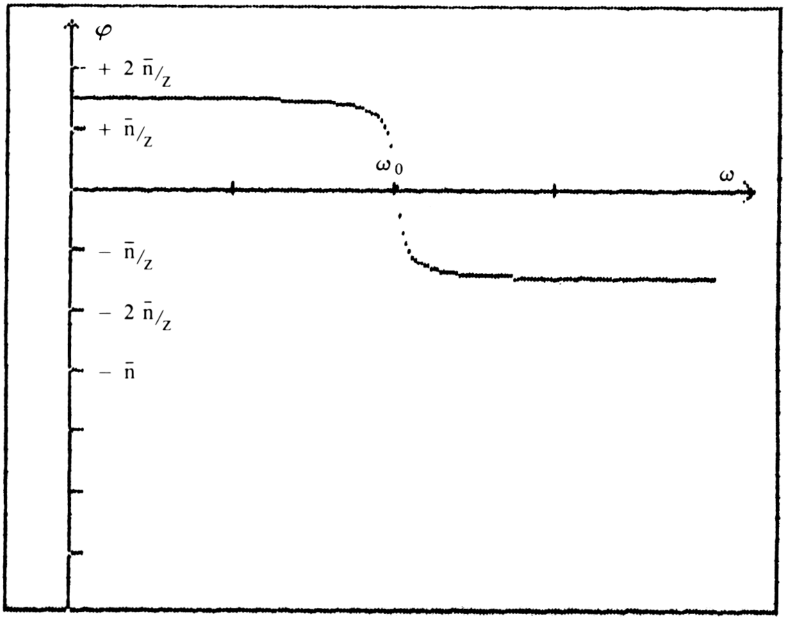
Filtre passe haut. m = 0,1. *Courbe de module. Étude sur 2 décades.* $T_R = 14$ dB.



Filtre passe haut. Courbe de phase. m = 0,1. *Étude sur 2 décades.*



Filtre passe bande. m = 0,05. *Étude sur 2 décades. Courbe de module.*



Filtre passe bande. m = 0,05. *Étude sur 2 décades. Courbe de phase.*

2. ÉQUATION DIFFÉRENTIELLE LINÉAIRE DU PREMIER ORDRE

INITIATION AU PROBLÈME

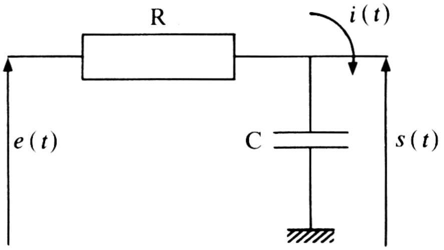
Il s'agit de tout système physique décrit par une équation du type :

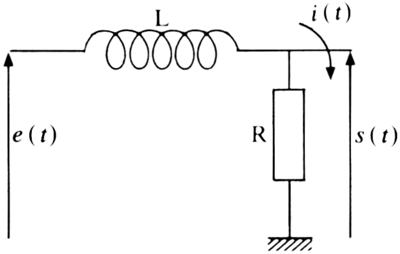
$$\frac{\tau dy(t)}{dt} + y(t) = Bx(t)$$

Les solutions d'une telle équation sont parfaitement connues pour des signaux $x(t)$ simples : échelon, sinusoïde. Dans les autres cas, il faudra une intégration numérique.

$$y(t) = y_0 \exp\left(\frac{-(t-t_0)}{\tau}\right) + \frac{B}{\tau}\int_{t_0}^{t} x(u)\exp\left(\frac{(u-t)}{\tau}\right)du$$

(Voir le programme Intégration.) Cette équation est celle qui régit le fonctionnement des circuits dits du premier ordre en électronique :





$$i(t) = \frac{e(t) - s(t)}{R}$$

$$i(t) = C\,\frac{ds(t)}{dt}$$

D'où :

$$RC\,\frac{ds(t)}{dt} + s(t) = e(t)$$

$$e(t) - s(t) = L\,\frac{di(t)}{dt}$$

$$i(t) = \frac{s(t)}{R}$$

D'où :

$$\frac{L}{R}\,\frac{ds(t)}{dt} + s(t) = e(t)$$

De façon générale :

$$\tau\,\frac{ds(t)}{dt} + s(t) = Be(t)$$

τ = constante de temps du système (en secondes).

B = transmittance statique du système.

MÉTHODES

Nous proposons de remplacer les dérivées par les rapports d'accroissements finis. Ainsi :

$$\tau\,\frac{\Delta s}{\Delta t} + s = Be$$

Appelons :

Δt = intervalle de calcul;

s_k = valeur de s à l'instant $k \cdot \Delta t$;

e_k = valeur de e à l'instant $k \cdot \Delta t$;

s_{k-1} = valeur de s à l'instant $(k-1)\,\Delta t$.

Ainsi :

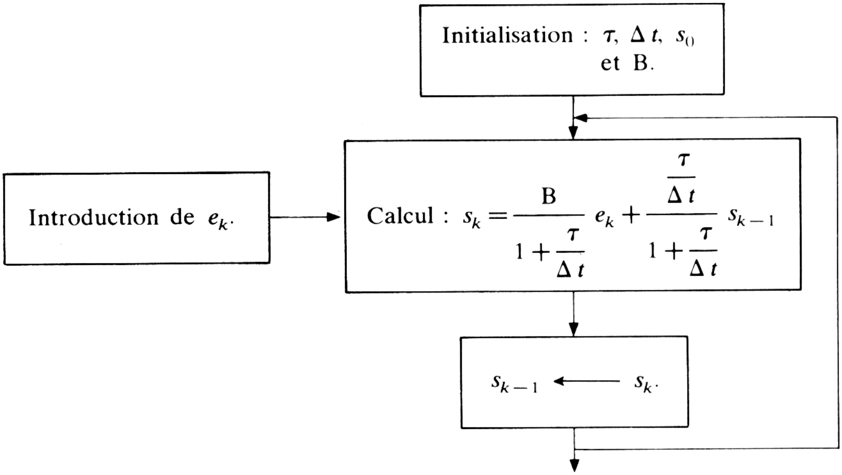
$$\tau \frac{s_k - s_{k-1}}{\Delta t} + s_k = \mathrm{B} e_k .$$

D'où :

$$s_k \left[1 + \frac{\tau}{\Delta t} \right] = \mathrm{B} e_k + \frac{\tau}{\Delta t} s_{k-1} .$$

Ainsi, la valeur *actuelle* de s_k peut être prédite, si l'on connaît sa valeur antérieure s_{k-1} et la valeur actuelle de l'entrée e_k. Pour commencer le calcul, il faut naturellement une condition initiale, qui est en général s_0 (ce qui sera par exemple la tension aux bornes du condensateur à $t = 0$). On en déduit facilement l'organigramme.

Remarquons que $e_k = e(k \Delta t)$ peut être soit une fonction explicite du temps : $e_k = \sin(k \Delta t)$, soit une fonction quelconque définie par un tableau (matrice) : $e_k = \mathrm{E}(k)$.



Pour les connaisseurs, il va de soi que Δt doit être *petit* devant τ qui caractérise la vitesse de réaction du système. Mais qu'en est-il si e est une fonction du temps? Nous reviendrons sur cette question plus loin.

L'erreur systématique à chaque pas est de l'ordre de $(\Delta t)^2$; l'erreur globale après sommation est un infiniment petit de l'ordre de Δt.

AMÉLIORATION DE LA MÉTHODE

La méthode précédente est simple. Essayons de la cerner avec plus de rigueur.

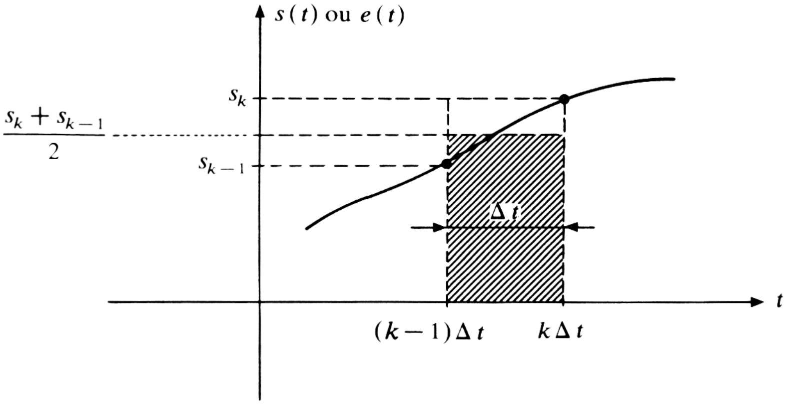
Reprenons l'équation différentielle du système :

$$\tau \frac{ds(t)}{dt} + s(t) = \mathrm{B}e(t)$$

Intégrons-la entre $(k-1)\Delta t$ et $k\Delta t$:

$$\int_{(k-1)\Delta t}^{k\Delta t} \tau \frac{ds(t)}{dt}\, dt + \int s(t)\, dt = \mathrm{B} \int e(t)\, dt$$

Les deux dernières intégrales peuvent s'évaluer avec la méthode du trapèze :



Il est facile de voir que l'aire correspondant à l'intégrale $\int s(t)\, dt$ peut être approchée par celle du rectangle hachuré :

$$\Delta t \frac{s_k + s_{k-1}}{2}$$

D'où :

$$\tau(s_k - s_{k-1}) + \Delta t \frac{s_k + s_{k-1}}{2} = \mathrm{B} \frac{e_k + e_{k-1}}{2}$$

et donc :

$$s_k = \frac{\tau - \frac{\Delta t}{2}}{\tau + \frac{\Delta t}{2}} s_{k-1} + \frac{\frac{\mathrm{B}}{2}}{\tau + \frac{\Delta t}{2}} (e_k + e_{k-1})$$

Cette relation, légèrement différente de la précédente, fait intervenir e_{k-1}.

Que pouvons-nous penser de la précision de la méthode et du choix optimal de Δt?

Pour cela, utilisons cette relation dans un cas où la solution de l'équation est parfaitement connue : $e(t)$ sinusoïdal. En notation complexe, on sait que la transmittance de tels systèmes est du type passe bas :

$$\underline{\mathrm{T}} = \frac{\underline{s}}{\underline{e}} = \frac{\mathrm{B}}{1 + j\omega\tau}$$

Posons alors :

$$\underline{e}_k = \mathrm{E}_0 \exp(j\omega k \Delta t) = \underline{\mathrm{E}} \exp(j\omega k \Delta t)$$

$$\underline{s}_k = \mathrm{S}_0 \exp(j\varphi) \exp(j\omega k \Delta t) = \underline{\mathrm{S}} \exp(j\omega k \Delta t)$$

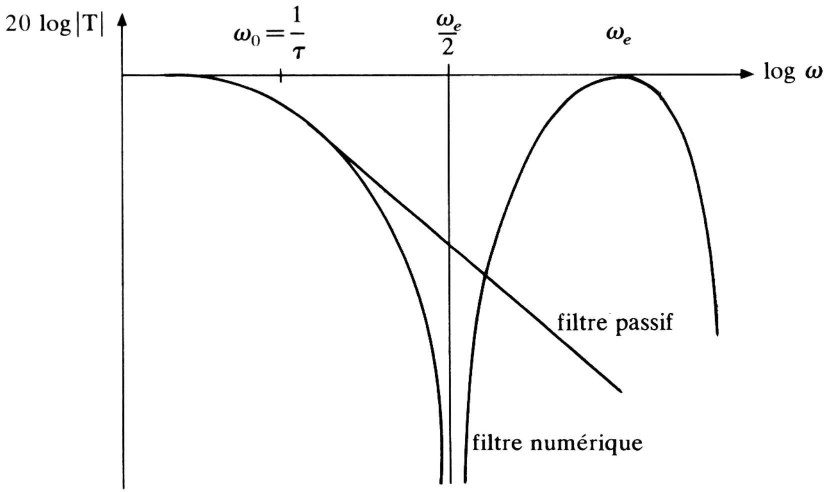
$$\underline{s}_{k-1} = \mathrm{S}_0 \exp(j\varphi) \exp(j\omega (k-1) \Delta t) = \underline{\mathrm{S}} \exp(j\omega (k-1) \Delta t)$$

(La solution, en régime permanent, est en effet sinusoïdale.)

Le développement de l'expression obtenue précédemment donne alors :

$$\underline{\mathrm{T}} = \frac{\underline{\mathrm{S}}}{\underline{\mathrm{E}}} = \frac{\mathrm{B}}{1 + \frac{j \operatorname{tg}\left(\frac{\omega \Delta t}{2}\right)}{\frac{\Delta t}{2\tau}}}$$

Si l'on compare les courbes donnant les modules des fonctions T par rapport à ω, on obtient :



Ces deux courbes sont apparemment différentes; cependant, on constate qu'elles coïncident pour les pulsations « petites » :

$$\operatorname{tg}(x) \simeq x; \; x = \frac{\omega \Delta t}{2} \cdot$$

Cela nous amène à nous interroger sur le sens physique de la méthode numérique étudiée. En effet, du point de vue de l'électronicien, il y a eu un échantillonnage à intervalle de temps Δt, donc avec la pulsation :

$$\omega_e = \frac{2\pi}{\Delta t}$$

Or (théorème de Shannon), cela n'a de sens que si la plus haute pulsation traitée ω est inférieure ou égale à la moitié de la pulsation de l'échantillonnage ω_e :

$$\omega \leqslant \frac{1}{2}\,\omega_e \quad \text{soit} \quad \omega \leqslant \frac{2\pi}{2\Delta t} = \frac{\pi}{\Delta t}$$

et donc la condition cherchée :

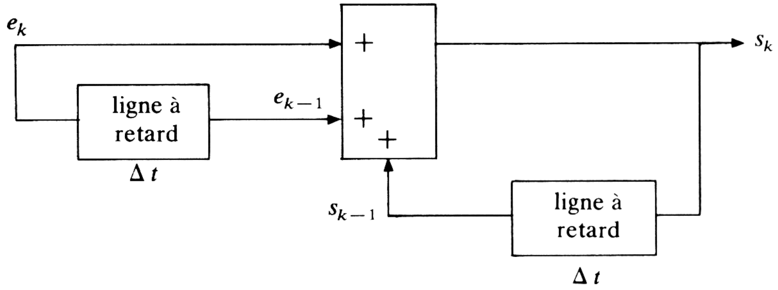
$$\Delta t \leqslant \frac{\pi}{\omega}$$

En pratique, on prendra un coefficient de sécurité. Les résultats obtenus par l'intégration numérique ne seront alors en théorie valables que si le développement en série de Fourier de $e(t)$ ne comprend pas d'harmoniques de pulsations supérieures à $\frac{\pi}{\Delta t}$. Pour que cela ait un sens, il faudra donc en général avoir :

$$\omega_0 \ll \frac{\pi}{\Delta t}$$

Remarque

Cette méthode n'est plus approchée, mais exacte, si l'on veut simuler le fonctionnement d'un filtre numérique, construit suivant la disposition :



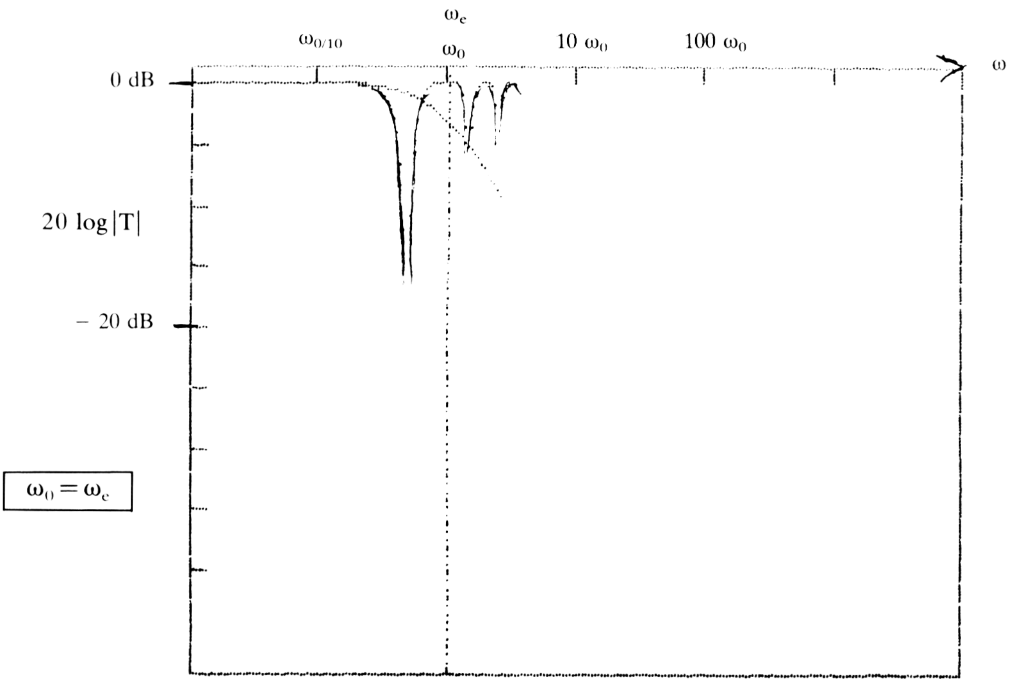
Il pourrait sembler que la méthode est d'autant plus précise que le *pas* de calcul est petit. Il n'en est rien. En effet, il n'est pas impossible, si ce pas est effectivement petit, qu'un calcul intermédiaire conduise à une variation d'une quantité plus petite que la précision du micro-ordinateur. Celle-ci est, dans les cas courants, de neuf chiffres. Ainsi : $1 + 10^{-10} = 1$; une variation de 10^{-10} ne peut être prise en compte pour une variable qui vaut 1.

APPLICATIONS

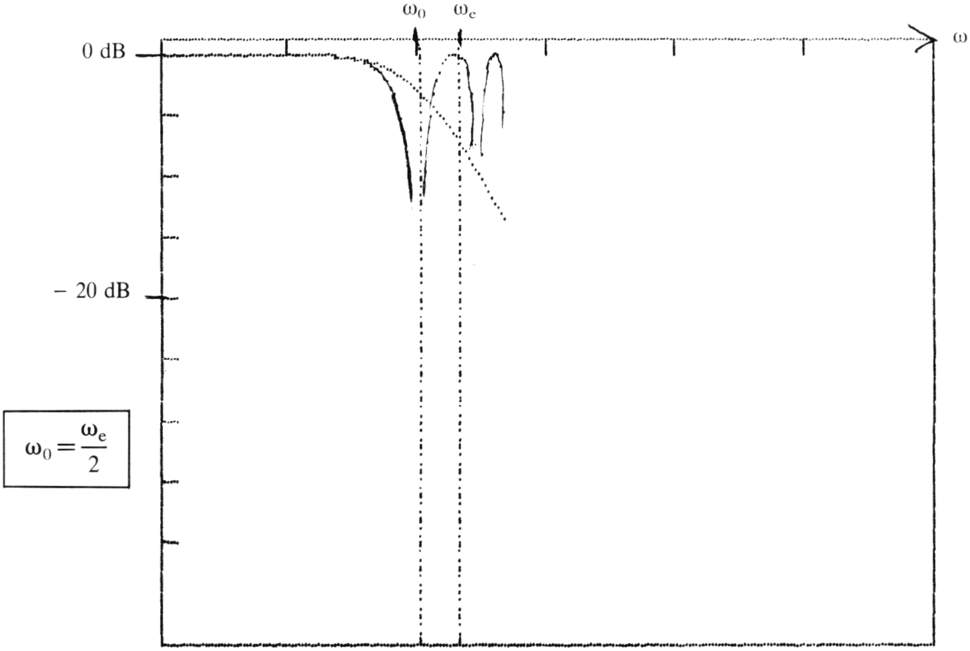
Filtre numérique

Comme conséquence directe de notre étude, nous pouvons essayer de comparer les courbes de réponse de filtres numériques et de filtres passifs, ayant même pulsation naturelle ω_0, en fonction de la fréquence d'échantillonnage ω_e. Le programme (« Filtrage ») est simple. Il permet de tracer les courbes en coordonnées logarithmiques.

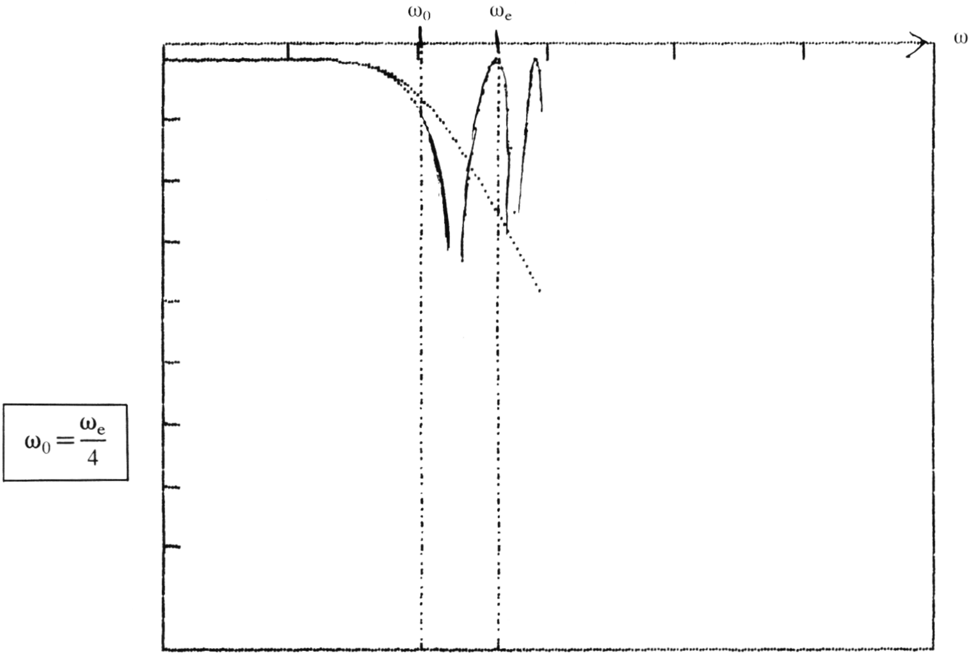
De façon générale, le filtre numérique se caractérise par une suite de nombreux zéros de transmission. En basse fréquence, sa courbe de réponse « colle » à celle du filtre passif; le domaine où les deux courbes coïncident est d'autant plus grand que ω_e est grand.



20 W0 = 2*PI/DT

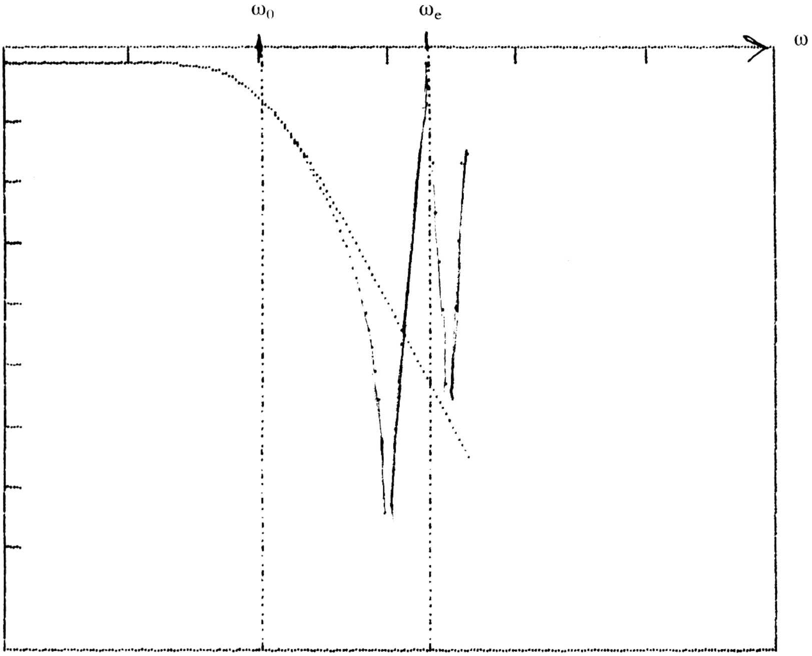


20 W0 = PI/DT



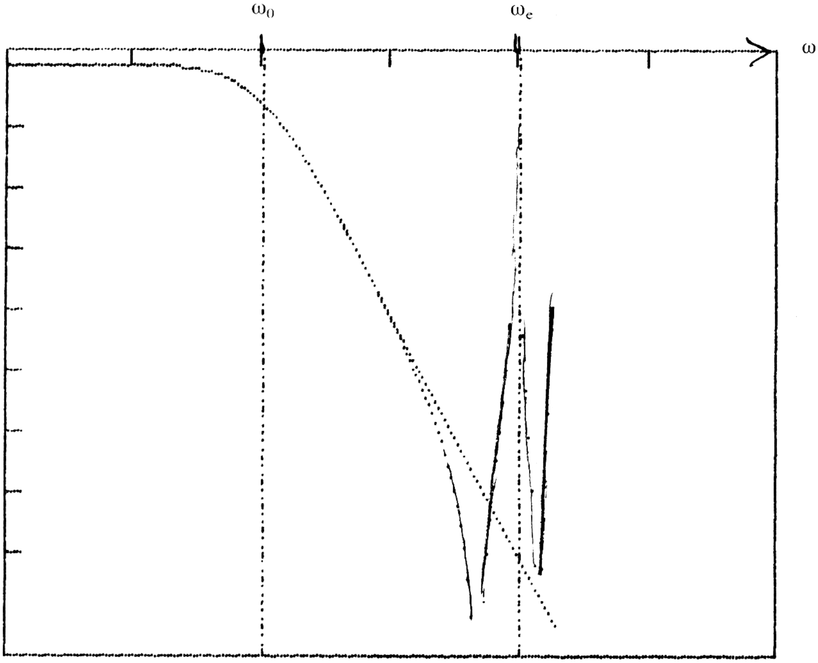
20 W0 = PI/(2*DT)

$$\omega_0 = \frac{\omega_e}{20}$$



20 W0 = PI/(10*DT)

$$\omega_0 = \frac{\omega_e}{100}$$



20 W0 = PI/(50*DT)

Programme "Filtrage"

```
 10 DT=1E-3
 20 W0=PI/(50*DT)
 30 W=W0/100
 40 HIRES
 50 GOSUB 200
 60 DEF FN F(W)=1/SQR(1+((W/W0)^2))
 70 DEF FN G(W)=1/SQR(1+((TAN(W*DT/2)/(W
0*DT/2))^2))
 80 N=10^(1/40)
 90 FOR U=0 TO 239
 100 W=W*N
 120 F=-80*LOG(FN F(W)): G=-80*LOG(FN G(
W))
 130 IF F>190 THEN 150
 140 CURSETU,F+5,1
 150 IF G > 190 THEN 170
 160 CURSETU,G+5,1
 170 NEXT U
 180 STOP
 200 REM AXES
 210 CURSET0,0,1:DRAW 239,0,1
 220 FOR N=1 TO 6
 230 CURSET N*40-1,0,1:DRAW0,5,1:NEXTN
 240 CURSET0,0,1:DRAW0,199,1:DRAW239,0,1
:DRAW0,-199,1
 250 FOR N=1 TO 8:CURSET0,20*N+5,1:DRAW5
,0,1:NEXTN
 260 PATTERN 45:CURSET40*LOG(2*PI/(DT*W0
))+80,0,1:DRAW0,199,1
 270 CURSET80,0,1:DRAW0,199,1
 280 RETURN
```

Charge d'un condensateur

L'équation du système a été donnée dans l'introduction du chapitre :

$$RC\,\frac{ds(t)}{dt} + s(t) = e(t)$$

Posons $e(t) =$ constante (cas de l'échelon de tension), $s(0) = S0$.

Le programme s'en déduit facilement (voir programme « Circuirc »). Quelques remarques cependant. Le raisonnement précédent ne s'applique pas à l'échelon, qui est un signal non périodique, et donc pour lequel le spectre de Fourier contient toutes les fréquences. Une étude graphique montre alors que, dans ce cas :

$$\frac{DT}{RC} \quad \text{soit} \quad \frac{dt}{\tau} \leqslant \frac{1}{10}$$

semble convenir.

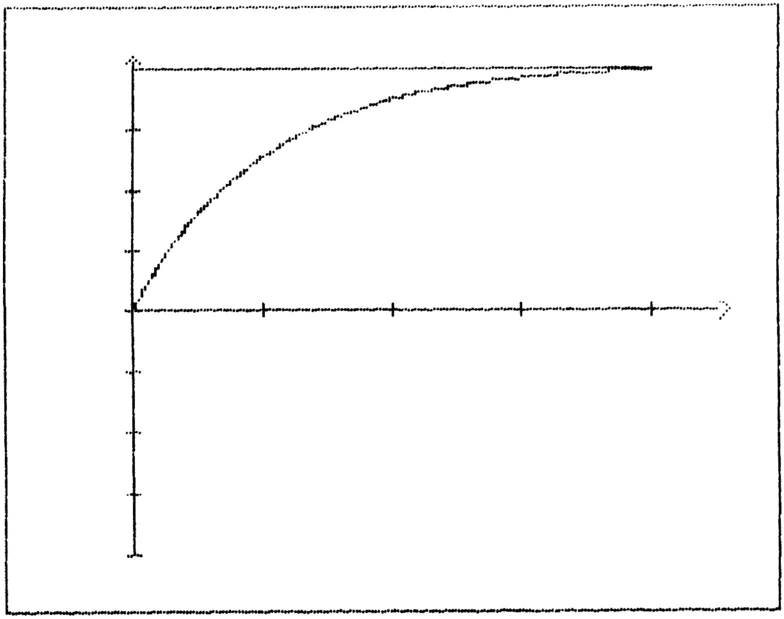
Par ailleurs, des valeurs trop grandes pour dt donnent des solutions plus *grandes* qu'en réalité. Cela vient du fait que la dérivée initiale de la fonction est prise en compte plus longtemps. Des valeurs trop petites pour dt ont pour conséquence une erreur d'arrondi. Un sous-programme (300) donne la solution exacte pour le cas où $s(0) = 0$.

Remarque

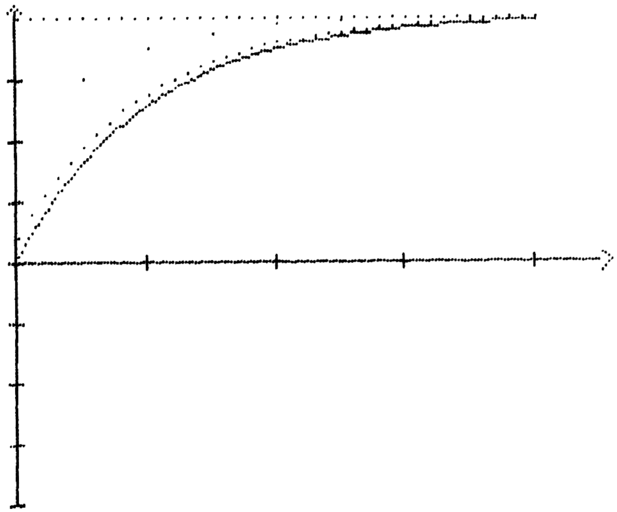
La solution théorique exacte est :

$$s(t) = s(0) \exp\left(\frac{-t}{RC}\right) + \frac{\text{Const}}{RC} \int_0^t \exp\left(\frac{(u-t)}{RC}\right) du$$

DT/RC = 1/100



s (0) = 0
e (t) = E *constante*
Les 2 courbes (solution numérique) et (solution exacte) sont superposées.



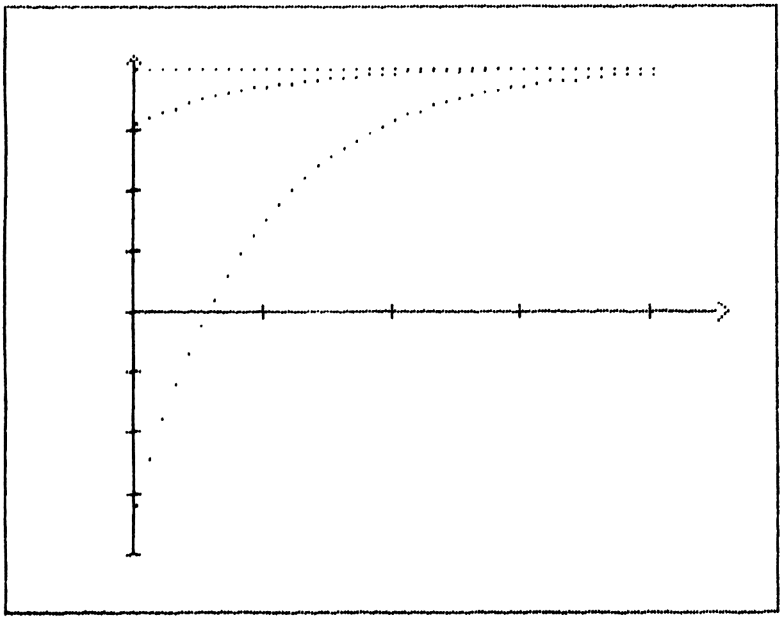
3 courbes superposées

— $\frac{DT}{RC} = \frac{1}{2}$

— $\frac{DT}{RC} = \frac{1}{10}$

— *solution exacte*

s (0) = 0
e (t) = E = *constante*



$\frac{DT}{RC} = \frac{1}{10^e}$

2 courbes, l'une pour s(0) = −10, *l'autre pour* s(0) = +7; *on a* e(t) = 10.

Programme "Circuirc"

```
20 INPUT"DT en fraction de RC:";DT
30 INPUT"Tension s initiale:";S
40 INPUT"Amplitude de l'echelon:";E
50 HIRES
60 GOSUB 200
70 FOR U=0 TO 160 STEP (40*DT)
80 DS=(E-S)*DT:S=S+DS
90 CURSET41+U,100-80*S/E,1
100 CURSET41+U,100-80*E/ABS(E),1
110 NEXTU
120 STOP
200 REM AXES
210 CURSET40,100,1:DRAW 180,0,1:CURMOV0
,-3,0:CHAR62,0,1
```

```
 220 CURSET40,180,1:DRAW 0,-160,1:CURMOV
-2,-4,0:CHAR94,0,1
 230 FORN=1TO 4:CURSET40+40*N,102,1:DRAW
0,-4,1:NEXTN
 240 FOR N=1 TO 8: CURSET 38,20+20*N,1:D
RAW4,0,1:NEXTN
 250 CURSET0,0,1:DRAW0,199,1:DRAW239,0,1
:DRAW0,-199,1:DRAW-239,0,1
 260 RETURN
 300 FOR U=0 TO 160
 310 CURSET40+U,(1-EXP(-U/40))*(-80)+100
,1
 320 NEXTU
 330 RETURN
```

Autre exemple

La même méthode peut servir à résoudre d'autres problèmes. Considérons le cas de la recherche de lignes équipotentielles, c'est-à-dire de lignes le long desquelles une fonction donnée F(X, Y) garde une valeur constante C. Le programme proposé montre l'exemple (ligne 600) de la fonction :

$$F(X, Y) = X^3 - Y^3 + 3XY$$

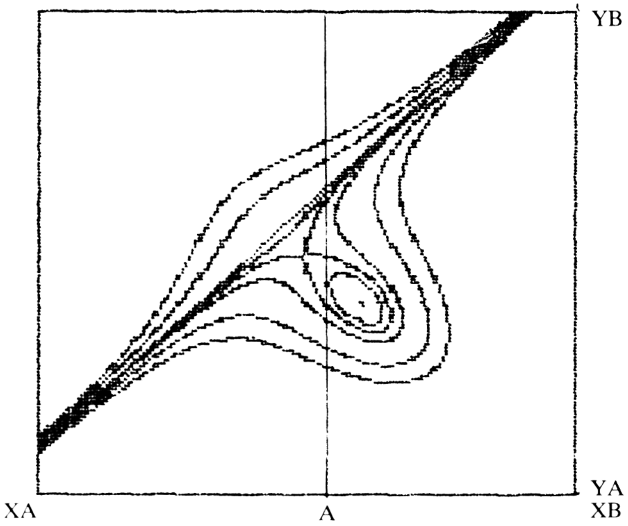
Il demande d'abord les valeurs extrêmes de X et de Y, ainsi que l'intervalle de calcul H (soit un Δt), qui est normalisé par la suite. Ensuite, il est demandé une valeur particulière de C, ainsi qu'une abscisse de départ A (POKE28,0 permet d'écrire dans les trois lignes de texte, en bas de l'écran haute résolution).

La méthode de Newton cherche alors un point initial de coordonnées A et Y(A), s'il existe. La courbe est décrite par les méthodes précédentes; elle est interrompue dès la détection d'une tangente verticale. La courbe est enfin reprise du même point, dans l'autre sens.

```
 10 REM *** POTENTIEL ***
 20 INPUT " x min = ";XA
 30 INPUT " x max = ";XB
 40 INPUT " y min = ";YA
 50 INPUT " y max = ";YB
 60 IF XB<=XA OR YB<=YA THEN PING: GOTO
30
 70 INPUT " Intervalle de calcul = ";H
 80 DIM Y(10), U(21)
 85 FOR K=0 TO 10
 90 Y(K)=1E10
 95 NEXT K
 100 HIRES
 110 POKE 28,0:REM Permet d'ecrire dans
les 3 lignes en bas de l'ecran Haute Res
 120 INPUT " f(x,y) = c ,c = ";C
 130 INPUT " Abscisse initiale a = ";A
 140 IF A>=XB OR A<=XA THEN PRINT " x mi
n < a < x max ": GOTO 130
 160 FOR K = 10 TO 0 STEP -1
 170 X=A : Y= YA + (YB-YA)*K/10
 180 GOSUB 500
 190 NEXT K
 200 IF J=0 THEN PRINT " Pas de solution
 f(a,y) = c":  GOTO 120
 210 GOSUB 800
 220 IF J=0 THEN PRINT " Pas de solution
 f(a,y) = c":  GOTO 120
 230 K=2
 235 IF K>2*J+1 THEN 320
 240 X=A: Y=U(K)
 250 GOSUB 700
 260 GOSUB 620: GOSUB 650
 270 DX=-FY/SQR(FX^2+FY^2):DY=FX/SQR(FX^
2+FY^2):X=X+H*DX:Y=Y+H*DY
 280 R=FY:GOSUB 650
 290 IF FY*R <=0 THEN H=-H :K=K+1:GOTO 2
35
 300 GOSUB 700
 310 IF X% = 0 OR X% =200 OR Y% = 0 OR Y
% =160 THEN H =-H  : K=K+1: GOTO 235
 315 GOTO 260
```

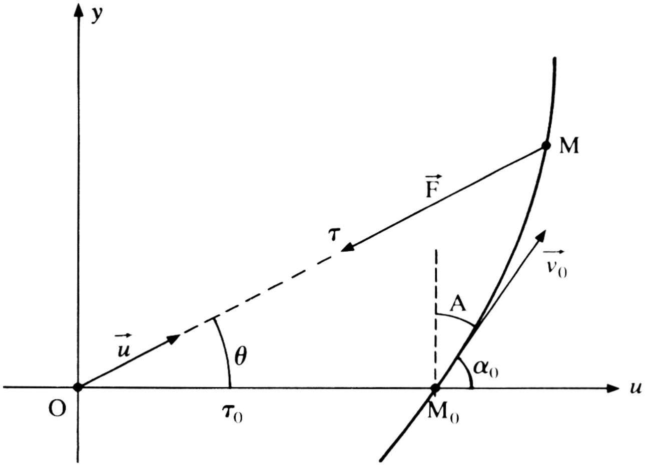
```
320 J=0: FOR K= 0 TO 10
330 Y(K)=1E10:U(2*K) =0: U(2*K+1)=0
340 NEXT K
350 GOTO 120
500 FOR N = 1 TO 100
510 GOSUB 600: GOSUB 650
515 IF FY = 0 THEN RETURN
520 Y1 = Y - F/FY
530 IF ABS(Y-Y1)<1E-3 THEN  J=1:Y(K)=Y1
: RETURN
540 Y=Y1
550 NEXT
560 RETURN
590 REM *** FONCTIONS F, Fx, Fy ***
600 F = X*X*X -Y*Y*Y + 3*X*Y -C
610 RETURN
620 XH=X: X=XH+H/100:GOSUB 600:FH=F
630 X=XH-H/100:GOSUB 600:FX=(FH-F)*50/H
:X=XH
640 RETURN
650 YH=Y: Y=YH+H/100:GOSUB 600:FH=F
660 Y=YH-H/100:GOSUB 600:FY=(FH-F)*50/H
: Y=YH
670 RETURN
690 REM *** TRACE ***
700 X% = (X-XA)*200/(XB-XA) : Y%=(YB-Y
)*160/ (YB-YA)
710 IF Y% > 160 THEN Y% = 160
720 IF Y% <0 THEN Y% =0
730 IF X% > 200 THEN X% = 200
740 IF X% < 0 THEN X% = 0
750 CURSET X%,Y%,1
760 RETURN
790 REM *** Valeurs initiales y(a) ***
800 K=0: J=0: IF Y(0)>YB OR Y(0)< YA TH
EN 820
810 J=J+1: U(2*J)=Y(K):U(2*J+1)=Y(K)
820 K=K+1: IF K=11 THEN RETURN
830 IF Y(K)>YB OR Y(K)<YA THEN 820
840 FOR L=0 TO K-1
850 IF (Y(K) > Y(L) - 1E-3 ) AND (Y(K)
< Y(L) + 1E-3 ) THEN 820
```

```
860 NEXT L
870 GOTO  810
```



3. MOUVEMENT PONCTUEL NEWTONIEN

Un mobile ponctuel, soumis à une force centrale centripète d'intensité inversement proportionnelle au carré de la distance, décrit un mouvement *newtonien.*



$\tau = OM$
$\vec{u}$: vecteur unitaire

Intéressons-nous au cas d'un satellite de la terre, que nous assimilerons à un objet ponctuel. On peut écrire :

$$\vec{F} = m\vec{a} = -G\frac{mm_T}{\tau^2}\vec{u}$$

Cette équation conduit à un mouvement plan.

En appelant $G(=6{,}67 \times 10^{-11}$ MKSA) la constante de gravitation universelle, on aboutit à :

$$\vec{a} = -\frac{Gm_T}{\tau^2}\vec{u}$$

Il s'agit d'un système d'équations différentielles, que l'on peut écrire :

$$\frac{d^2 x}{dt^2} = -\mathrm{G} m_{\mathrm{T}} \cdot \frac{x}{(x^2 + y^2)^{3/2}}$$

$$\frac{d^2 y}{dt^2} = -\mathrm{G} m_{\mathrm{T}} \cdot \frac{y}{(x^2 + y)^{3/2}}$$

La résolution de ce système dépend de deux conditions initiales (à $t = 0$); on se donne en général $\vec{r_0}$ (distance au centre O) et la vitesse de satellisation $\vec{v_0}$ (module et angle de départ).

Il y a (au moins) deux méthodes pour résoudre ce problème :

- Une résolution directe du système d'équations, en étudiant l'évolution du système pendant des incréments de temps *dt*. Cette méthode a l'avantage d'utiliser ainsi le temps comme paramètre, ce qui permet de suivre le satellite en *temps réel*.
- Une solution analytique du phénomène, c'est-à-dire le tracé direct de la trajectoire. Le temps n'intervient alors pas.

PREMIÈRE MÉTHODE

Elle est classique et a déjà été étudiée dans cet ouvrage. Nous posons $\mathrm{K} = \mathrm{G} m_{\mathrm{T}}$. L'intervalle de temps choisi est DT = 10 secondes, ce qui paraît raisonnable. Les conditions initiales sont : l'altitude d'éjection H, l'angle d'éjection A et la vitesse d'éjection V0.

Le programme évalue dans un premier temps (à titre indicatif pour le lecteur) la plus petite vitesse permettant la satellisation dans le cas :

$$\alpha_0 = \frac{\pi}{2}$$

Un facteur d'échelle F est calculé, prenant le rayon de la terre R_0 comme référence si G = 1. 50 points son l'écran représentent R_0. Si G = 0,5, cette échelle est divisée par deux.

Les grandeurs initiales en coordonnées cartésiennes sont ensuite évaluées.

Le système d'équations différentielles est enfin intégré. Trois exemples de courbes obtenues par ce programme sont fournis; on notera que, dans le troisième cas, le satellite s'écrase sur la terre!

```
 10 K=9.8*(6.8^2)*1E12
 20 INPUT"ALTITUDE D'EJECTION (en m )=";
H
 30 DT=10:R0=6.8E6
 40 VM=SQR(2*K*R0/((R0+H)*(2*R0+H)))
 50 PRINTVM
 60 INPUT"VITESSE D'EJECTION (en m/s )="
;V0
 70 INPUT"ANGLE D'EJECTION (en degres )=
";A
 80 INPUT"FACTEUR D'ECHELLE =";G
 90 F=G*50/6.8E6
 100 R=R0+H:VY=V0*COS(A*2*PI/360):VX=V0*
SIN(A*2*PI/360):X=R0+H:Y=0
 110 HIRES
 120 CURSET120,100,1:CIRCLE50*G,1
 130 CURSET0,0,1:DRAW239,0,1:DRAW0,199,1
:DRAW-239,0,1:DRAW0,-199,1
 140 R3=R^3:AX=-K*X/R3:AY=-K*Y/R3
 150 VX=VX+AX*DT:VY=VY+AY*DT
 160 X=X+VX*DT:Y=Y+VY*DT
 170 CURSET 120+(X*F),100-(Y*F),1
 180 R=SQR(X*X+Y*Y)
 190 T1=T1+DT:T=INT(T1/3600):PRINTT;" h.
"
 200 PRINTT1
 210 GOTO140
 220 LPRINT"FACTEUR D'ECHELLE = ";G
 230 LPRINT"ALTITUDE DE DEPART";H;" m"
 240 LPRINT"VITESSE D'EJECTION";V0;" m/s
"
 250 LPRINT"ANGLE D'EJECTION";A;" degres
"
 260 LPRINT"VITESSE MINIMUM =";VM;" m/s"
 270 LPRINT"PERIODE DE REVOLUTION =";T1;
" s"
 280 STOP
```

SECONDE MÉTHODE

On démontre que la courbe décrite a pour équation :

$$\frac{1}{\tau}=\frac{G m_T}{C^2}+E \cos \theta+B \sin \theta$$

avec :

$$C=\tau_0\, v_0 \sin \alpha_0$$

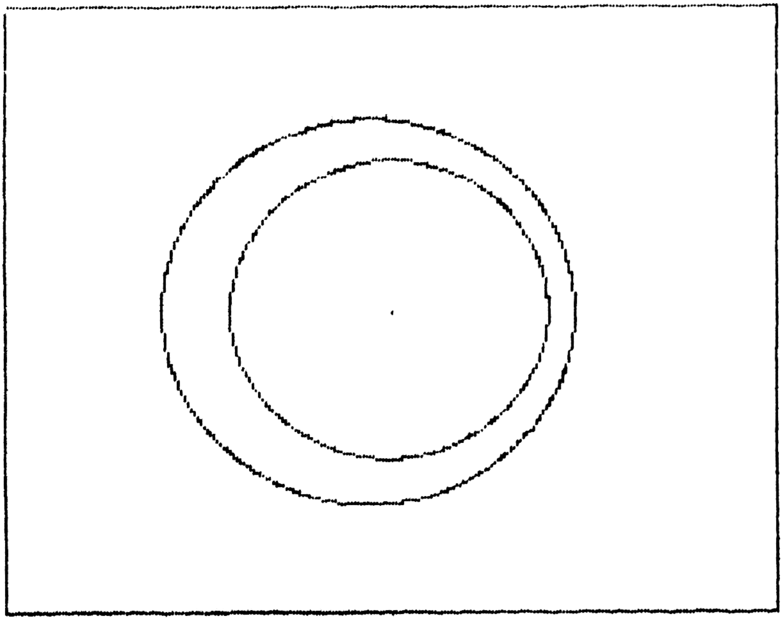
$$E=\frac{1}{\tau_0}-\frac{G m_T}{\tau_0^2\, v_0^2 \sin^2 \alpha_0}$$

$$B=-\frac{\text{Cotg}\ \alpha_0}{\tau_0}$$

Les conditions initiales à introduire sont les mêmes que précédemment : τ_0, v_0, A. S'agissant de la solution exacte du problème, la précision du résultat est liée au choix de l'incrément de l'angle θ; l'équation étant en effet du type polaire implicite $\tau=f(\theta)$. Les trois mêmes exemples ont été choisis, à fin de comparaison.

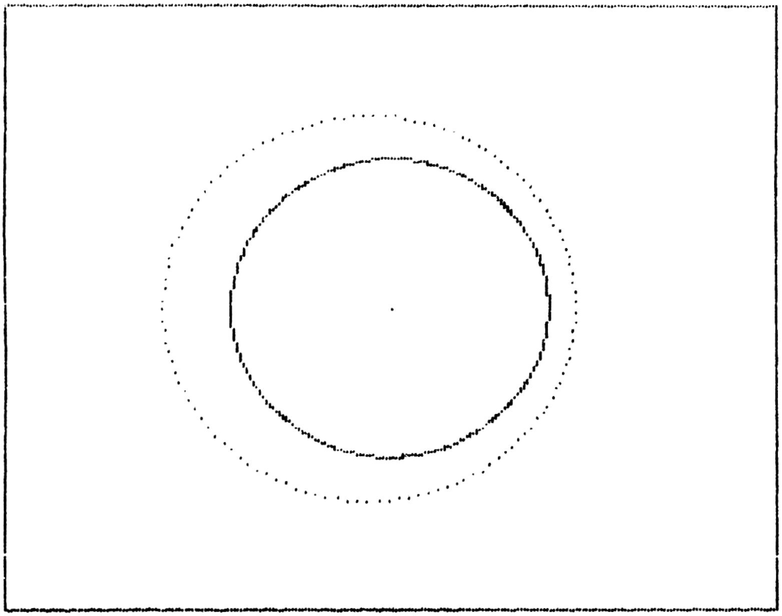
```
 10 K=9.8*(6.8^2)*1E12
 20 INPUT"ALTITUDE D'EJECTION (en m )=";
H
 30 DT=10:R0=6.8E6
 40 VM=SQR(2*K*R0/((R0+H)*(2*R0+H)))
 50 PRINTVM
 60 INPUT"VITESSE D'EJECTION (en m/s )="
;V0
 70 INPUT"ANGLE D'EJECTION (en degres )=
";A:A=90-A
 80 INPUT"FACTEUR D'ECHELLE =";G
 90 F=G*50/6.8E6
 100 R=R0+H:D=R*V0*SIN(A*2*PI/360):E=((1
/R)-(K/(D*D))):B=-1/(R*TAN(A*2*PI/360))
 110 HIRES
 120 CURSET120,100,1:CIRCLE50*G,1
```

```
130 CURSET0,0,1:DRAW239,0,1:DRAW0,199,1
:DRAW-239,0,1:DRAW0,-199,1
140 T=0:C2=D*D
150 T=T+.05
160 R=1/((K/C2)+(E*COS(T))+(B*SIN(T)))
170 X=F*R*COS(T):Y=F*R*SIN(T)
180 IF X>119 OR X<-119 OR Y>99 OR Y<-99
THEN 200
190 CURSET 120+X,100-Y,1:GOTO150
200 INPUT"Voulez vous continuer ";A$
210 IF A$ = "O" THEN GOTO 150 ELSE STOP
```

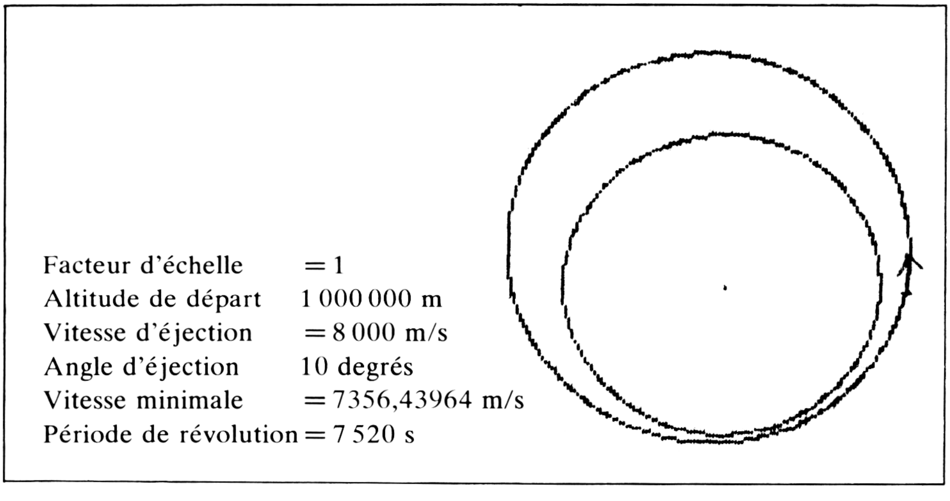


Programme 1

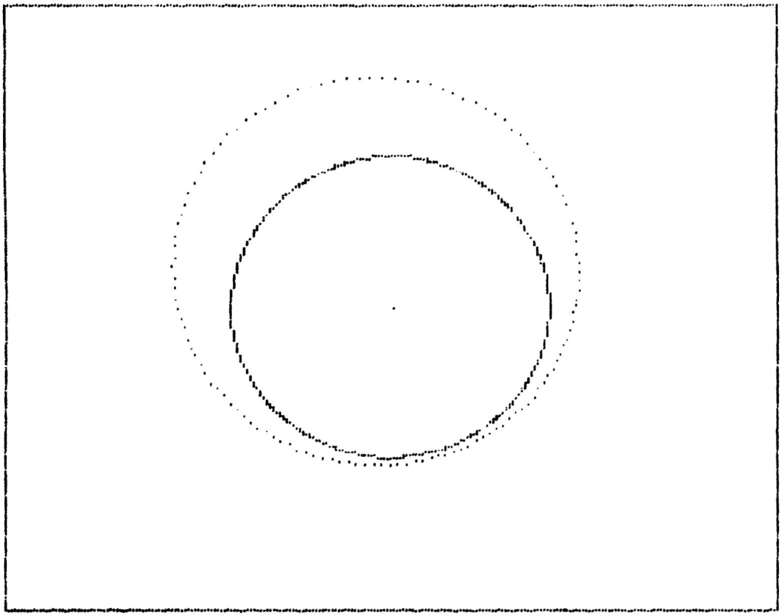
Facteur d'échelle	= 1
Altitude de départ	1 000 000 m
Vitesse d'éjection	= 8 000 m/s
Angle d'éjection	0 degrés
Vitesse minimale	= 7356,43964 m/s
Période de révolution	= 7 540 s



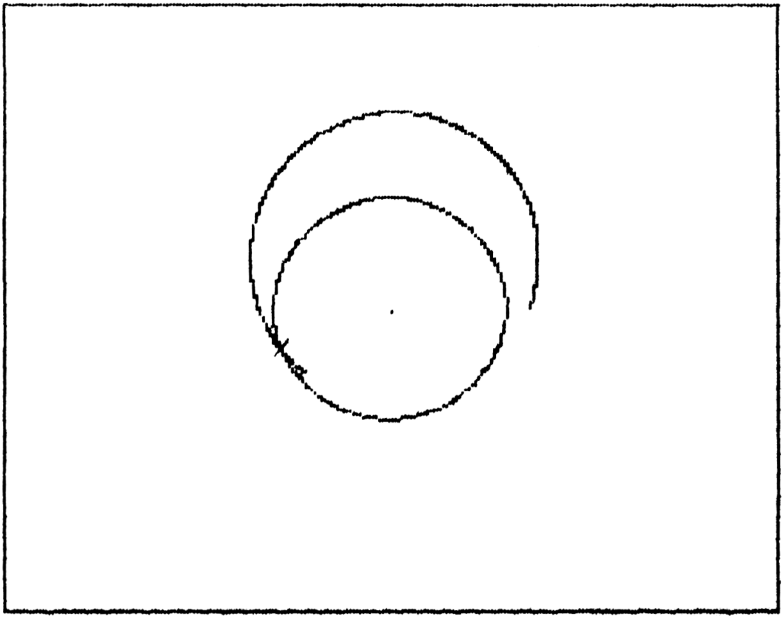
Programme 2



Programme 1

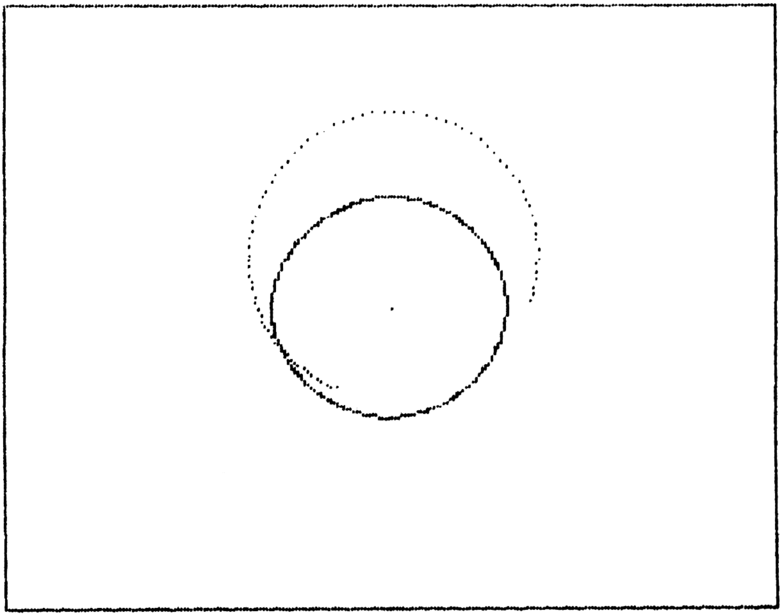


Programme 2



Programme 1

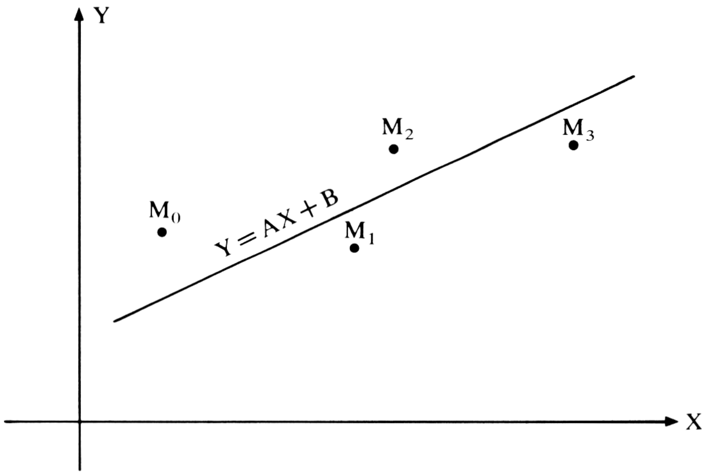
Facteur d'échelle	= 0,75
Altitude de départ	1 000 000 m
Vitesse d'éjection	= 8 000 m/s
Angle d'éjection	20 degrés
Vitesse minimale	= 7356,43964 m/s
Période de révolution	= 0 s



Programme 2

4. RÉGRESSION LINÉAIRE

Considérons un ensemble N de points, de coordonnées X et Y. Le problème est de trouver une droite passant entre tous ces points, de telle manière que la somme des carrés des distances des points à cette droite soit minimale (méthode des moindres carrés). Si tous les points sont alignés, il se trouvent évidemment sur cette droite. On dit alors que leur corrélation est parfaite. Inversement, dans le cas général, il est possible de définir un coefficient de corrélation *s* permettant de chiffrer dans quelle mesure ces points sont alignés. On démontre les relations suivantes :




```
 10 INPUT"Nombre de couples de Points:";
N
 20 SX=0:SY=0:X2=0:Y2=0:U=0
 30 FOR J=1 TO N
 40 PRINTJ,:INPUT"X,Y";X,Y
 45 LPRINTJ,X,Y
```

```
50 SX=SX+X
60 SY=SY+Y
70 X2=X2+X*X
80 Y2=Y2+Y*Y
90 U=U+X*Y
100 NEXT J
110 A=(N*U-SX*SY)/(N*X2-SX*SX)
120 B=(SY-A*SX)/N
130 PRINT:PRINT:PRINT
140 PRINT"Y(x) = ";A;" x + ";B
145 LPRINT:LPRINT:LPRINT"  Y(x) = ";A;"
x + ";B
150 V=A*(U-(SX*SY/N))
160 W=Y2-(SY*SY/N)
170 S=V/W
180 PRINT"Le coefficient de correlation
est:"
190 PRINT"               s = ";SQR(S)
195 LPRINT:LPRINT:LPRINT" Le coefficien
t de correlation est :";SQR(S)
200 PRINT:PRINT:PRINT
210 PRINT"Voulez-vous estimer un Y conn
aissant x?":GETA$:IFA$="N"THEN STOP
220 INPUT"x=";X
230 PRINT"Y = ";A*X+B
235 LPRINT:LPRINT:LPRINT" Y est egal a
";A*X+B;"Pour x = ";X
240 GOTO 210
```

En posant :

$SX = \text{somme des } X = \sum_{1}^{N} X_i$

$SY = \text{somme des } Y = \sum_{1}^{N} Y_i$

$X2 = \text{somme des carrés des } X = \sum_{1}^{N} X_i^2$

$$Y2 = \text{somme des carrés des } Y = \sum_{1}^{N} Y_i^2$$

$$U = \text{somme des produits } XY = \sum_{1}^{N} X_i Y_i$$

on obtient :

$$A = \frac{N \cdot U - (SX \cdot SY)}{N \cdot X2 - (SX)^2} \quad \text{et} \quad B = \frac{1}{N}(SY - A \cdot SX)$$

$$s^2 = \frac{A \cdot \left(U - \dfrac{SX \cdot SY}{N}\right)}{Y2 - \dfrac{(SY)^2}{N}}$$

EXEMPLE

Considérons l'ensemble des températures moyennes relevées à Marseille en septembre 1984. La coordonnée X représentera le quantième du jour concerné, la coordonnée Y, la température moyenne de ce même jour. La loi obtenue (voir le tableau) est $Y = -0{,}16\,X + 23{,}6$. Le coefficient de corrélation, cependant, n'est pas très bon : 0,66. (Cela est dû au fait que le temps a des « caprices » en cette saison de l'année.) On peut déduire de tout cela que :

- la température moyenne de septembre a été : 21,13 °C ($X = 15$);
- la température moyenne de septembre a décru de 23,43 °C ($X = 1$) à 18,66 °C ($X = 30$), soit une chute de 4,77 °C au cours du mois;
- la température moyenne d'octobre sera ($X = 45$) : 16,19 °C, cette prévision étant toutefois faite sous toutes réserves! (En fait, elle a été de 18,80 °C.)

1	1	24
2	2	25
3	3	24.5
4	4	25
5	5	22
6	6	19.5
7	7	19.5
8	8	20.5
9	9	21.5
10	10	22
11	11	22
12	12	22
13	13	21.5
14	14	22
15	15	23.5
16	16	22
17	17	23
18	18	20.5
19	19	18.5
20	20	21.5
21	21	21
22	22	19.5
23	23	18
24	24	17
25	25	17
26	26	18
27	27	21.5
28	28	20
29	29	20
30	30	19.5

Y(x) = -.164516129 x + 23.6

Le coefficient de correlation est : .663504074

Y est egal a 16.1967742 pour x = 45

Y est egal a 21.1322581 pour x = 15

```
Y est egal a  23.4354839 pour x =  1

Y est egal a  18.6645161 pour x =  30
```

II
LES FONCTIONS EN ANALYSE ET LEUR TRAITEMENT

5. QUELQUES FONCTIONS IMPORTANTES EN ANALYSE

INITIATION AU PROBLÈME

Un signal modulé en fréquence est de la forme :

$$u = \mathrm{U} \sin(\Omega_0 t + \varphi_0 + x \sin \omega t)$$

x indice de modulation.

Et si $\varphi_0 = 0$:

$$u = \mathrm{U}\{\sin \Omega_0 t \cos(x \sin \omega t) + \cos \Omega_0 t \sin(x \sin \omega t)\}.$$

On peut développer les termes :

$$\sin(x \sin \omega t), \cos(x \sin \omega t)$$

en une série de Fourier dont les coefficients sont les fonctions de Bessel de première espèce :

$$\mathrm{J}_0(x), \mathrm{J}_1(x), \ldots \mathrm{J}_n(x) = \sum_{k=0}^{\infty} \frac{(-1)^k}{\Gamma(n+k+1)\, k!} \left(\frac{x}{2}\right)^{2k+n}$$

où $\Gamma(x)$ désigne la fonction gamma :

$$\Gamma(n+k+1) = (n+k)!$$

On trouve finalement :

$$u = \mathrm{U}\{\sin \Omega_0 t (\mathrm{J}_0(x) + 2\,\mathrm{J}_2(x) \cos 2\,\omega t + 2\,\mathrm{J}_4(x) \cos 4\,\omega t + \ldots)$$
$$+ \cos \Omega_0 t (2\,\mathrm{J}_1(x) \sin \omega t + 2\,\mathrm{J}_3(x) \sin 3\,\omega t + \ldots)\}$$

$$= \mathrm{U}\left(\mathrm{J}_0(x) \sin \Omega_0 t + \sum_{n=1}^{\infty} \mathrm{J}_n(x) \{\sin(\Omega_0 + n\omega)\, t + (-1)^n \sin(\Omega_0 - n\omega)\, t\}\right)$$

et l'on voit aisément une infinité d'ondes latérales dont les amplitudes sont liées aux fonctions de Bessel de l'ordre *n* et de l'indice de modulation *x*.

Les ondes latérales de rangs supérieurs à $\frac{3x}{2}$ sont négligeables.

La Figure 5.1 montre quelques fonctions de Bessel.

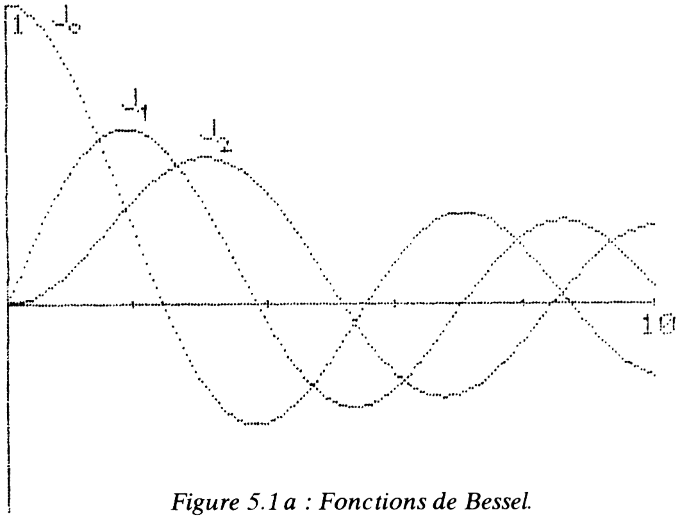


Figure 5.1 a : Fonctions de Bessel.

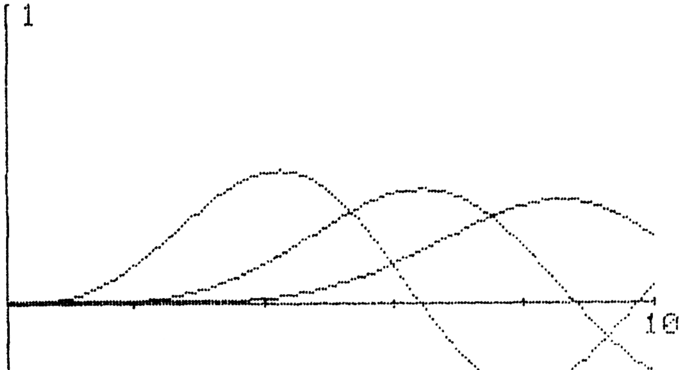


Figure 5.1 b : Fonctions de Bessel.

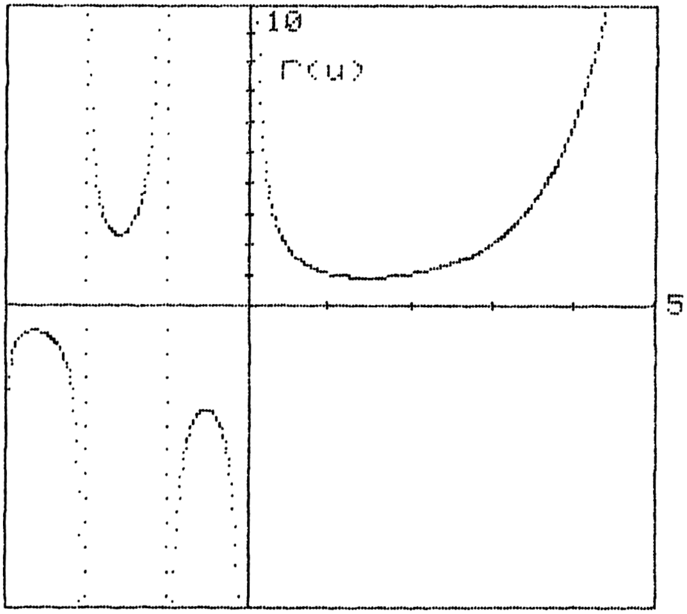


Figure 5.2 : Fonction gamma.

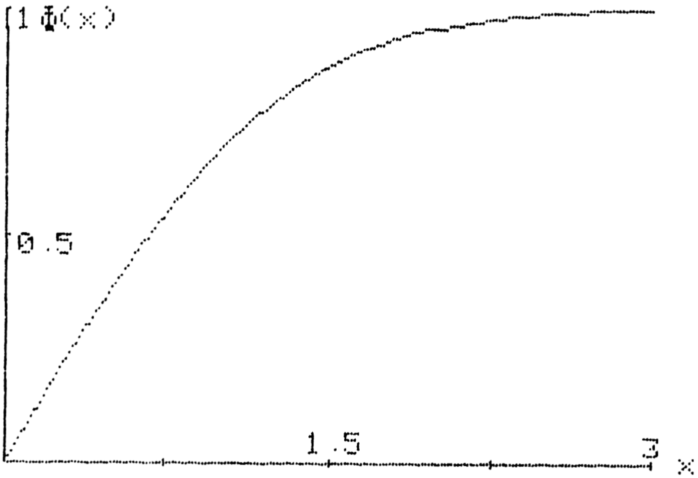


Figure 5.3 : Fonction d'erreur.

MÉTHODE

On propose donc plusieurs programmes qui servent à calculer les fonctions de Bessel et la fonction gamma.

Le premier programme est basé sur le développement en série :

$$J(x, n) = \frac{\left(\frac{x}{2}\right)^n \sum_{k=0}^{\infty} b_k}{\Gamma(n+1)}$$

$$b_0 = 1, \; b_k = \frac{-b_{k-1}\, x^2}{(4\,k\,(n+k))}, \; k = 1, 2, 3, \ldots$$

Il faut introduire le paramètre n, la valeur $\Gamma(n+1)$ si n n'est pas un nombre entier, et la valeur de la variable x, $0 < x < 7 + |n|$. La sommation est tronquée selon la précision voulue (sept chiffres). Le deuxième programme donne aussi les fonctions de Bessel de seconde espèce en se servant du développement asymptotique :

$$J_n(x) + jY_n(x) =$$

$$= \sqrt{\frac{2}{\pi x}}\, e^{j(x - \pi/2\, n - \pi/4)} \left\{ \sum_{m=0}^{k-1} \frac{j^m}{(2\,x)^m} \frac{\Gamma\left(m + n + \frac{1}{2}\right)}{m!\,\Gamma\left(m - n + \frac{1}{2}\right)} \right.$$

$$\left. + c\left(\frac{j}{2\,x}\right)^k \frac{\Gamma\left(k + n + \frac{1}{2}\right)}{k!\,\Gamma\left(k - n + \frac{1}{2}\right)} \right\} \; |c| < 1, \; n > -\frac{1}{2}$$

Pour $x \geqq 7 + |n|$, sept chiffres significatifs sont garantis.

Quand $2\,n$ est un nombre impair, la série ne possède qu'un nombre fini de termes. Dans les autres cas, on arrête la sommation dès que les termes recommencent à grandir.

Le troisième programme concerne la fonction gamma. Nous rappelons sa définition :

$$\Gamma(u)=\int_0^{\infty-t} e^{-t}\, t^{u-1}\, dt,\ u>0$$

C'est pourquoi une intégration par parties donne :

$$\Gamma(u+1)=u\Gamma(u),\ \Gamma(1)=1$$

et on déduit : $\Gamma(n+1)=n!$ pour tout n entier$=1, 2, 3, \ldots$

En se servant de la relation :

$$\Gamma(z)=\frac{\Gamma(z+m+1)}{(z+m)(z+m-1)\ldots(z+1)z}$$

on voit que la fonction possède un prolongement analytique dans le domaine :

$$\mathbb{C}/\{0,\ -1,\ -2,\ -3,\ \ldots\}$$

On a donc $\Gamma(n)=\infty$ pour $n=0,\ -1,\ -2,\ -3,\ \ldots$

Pour évaluer $\Gamma(u)$, on utilise son développement asymptotique :

$$\Gamma(u)\cong\sqrt{2\pi}\exp\left\{\left(u-\frac{1}{2}\right)\ln u-u+\frac{1}{12\,u}-\frac{1}{360\,u^3}+\frac{1}{1\,260\,u^5}-\frac{1}{1\,680\,u^7}+-\ldots\right\}(u\geqq 6)$$

On peut attendre sept à huit chiffres significatifs (Figure 5.2).
Un quatrième programme étudie la fonction d'erreur :

$$\Phi(x)=\sqrt{\frac{2}{\pi}}\int_0^x e^{-t^2/2}\,dt$$

Dans le but de calculer la distribution normale, on utilise sa densité de probabilité :

$$d(t) = \frac{1}{\sqrt{2\pi}} e^{-t^2/2}$$

La fonction d'erreur est donc étroitement liée à la loi normale (voir la Figure 5.3). Dans l'intervalle [0,3] on se sert de son développement en série :

$$\Phi(x) = \sqrt{\frac{2}{\pi}}\, x e^{-x^2/2} \left(1 + \frac{x^2}{3} + \frac{x^4}{3 \cdot 5} + \frac{x^6}{3 \cdot 5 \cdot 7} + \ldots\right)$$

pour $x > 3$, de sa fraction continue :

$$\Phi(x) = 1 - \frac{1}{\sqrt{\pi}} e^{-x^2/2} \left(\frac{1|}{|x} + \frac{1|}{|x} + \frac{2|}{|x} + \frac{3|}{|x} + \frac{4|}{|x} + \ldots\right)$$

On tronque les sommations après 23 (resp. 16) termes pour garantir huit chiffres significatifs [1].

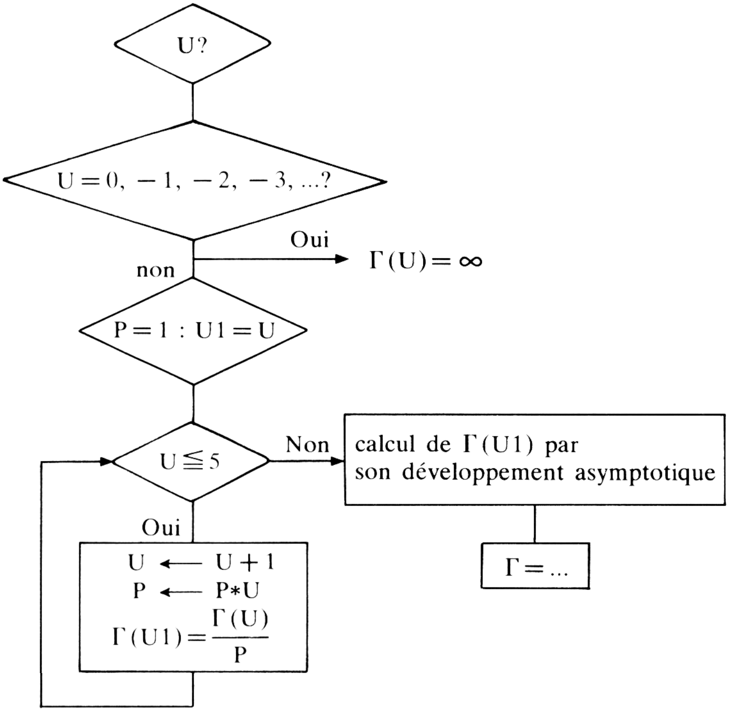
ORGANIGRAMME

Comme les méthodes des quatre programmes proposés se ressemblent, on ne donne qu'un seul organigramme concernant la fonction gamma.

---

1. *An a priori Estimate for the Truncation Error of a Continued Fraction Expansion to the Gaussian Error Function*, G. Boesse, Computing 29, 1982, p. 135-152.

Programme : Fonction gamma



LES PROGRAMMES

```
2200 REM *** Fonction d'erreur ***
2210 REM * I(SQR(2/PI)*EXP(-t^2/2),0,x)
*
2220 INPUT "Valeur de la variable x = "
;X
2230 IF X > 3 THEN 2300
2240 S=1:B=1
2250 FOR M=3 TO 45 STEP 2
2260 B = B*X*X/M: S=S+B
2270 NEXT
2280 S = S*X*SQR(2/PI)*EXP(-X*X/2)
```

```
2290 PRINT "I(SQR(2/PI)*EXP(-t^2/2),0,"
;X;") = ";S: END
2300 S=X
2310 FOR M=15 TO 1 STEP -1
2320 S = X + M/S
2330 NEXT
2340 S=1-1/SQR(PI)* EXP(-X*X/2)/(X+S)
2350 GOTO 2290


2400 REM *** Fonctions de BESSEL ***
2405 REM *** J(x,n), Programme 1 ***
2410 INPUT "Parametre n = ";N
2415 IF N<>INT(N) OR N<0 THEN 2445
2420 G=1: FOR K=0 TO N-1
2430 G=G*(K+1)
2435 NEXT
2440 GOTO 2455
2445 INPUT "Gamma(n+1) + ";G
2450 REM *** Voir Programme fonction  g
amma ***
2455 INPUT "Valeur de la variable x = "
;X
2460 REM *** 0<x<7+abs(n) ***
2465 IF (X<=0) OR (X>=7+ABS(N)) THEN PR
INT"x >= 7 + abs(n); voir Programme 2 "
2470 REM *** Algorithme ***
2480 B=1: M=1: J=B
2490 B = -B*X*X/(4*M*(N+M))
2500 J = J + B: M=M+1
2510 IF ABS(B)>1E-9  THEN 2490
2520 J=J*(X/2)^N/G
2530 PRINT "J(";X;",";N;") = ";J: END


2600 REM *** Fonctions de BESSEL ***
2605 REM *** J(x,n), Y(x,n) Progr. 2 **
*
2610 REM *** Developpement asymptotique
 x>>0 ***
2620 REM *** Precision: 7 chiffres ***
```

```
 2630 INPUT "Parametre n = ";N
 2640 INPUT "Valeur de la variable x = "
;X
 2650 IF X< 7+ABS(N) THEN PRINT "x trop
petit ": IF X<=0 THEN PING: GOTO 2640
 2660 REM *** Algorithme ***
 2700 B=1: M=1: R=1: I=0
 2710 C=B: B=B*(-.5+N+M)*(-.5-N+M)/(2*M*
X)
 2720 IF M=INT(M/2)*2 THEN R = R+(-1)^(M
/2)*B: GOTO 2740
 2730 I = I - (-1)^INT(M/2)*B
 2740 IF (B=0) OR((ABS(B)> ABS(C)) AND (
M> ABS (N))) THEN 2760
 2750 M=M+1: GOTO 2710
 2760 A= SQR(R*R + I*I)
 2770 P = ATN(I/R)
 2780 IF R< 0 THEN P = PI+P
 2790 J = SQR(2/(PI*X))*A*COS(X-N*PI/2-P
I/4+P)
 2800 Y = SQR(2/(PI*X))*A*SIN(X-N*PI/2-P
I/4+P)
 2810 PRINT "J(";X;",";N;") = ";J
 2820 PRINT "Y(";X;",";N;") = ";Y
 2830 END



 2900 REM *** Fonction GAMMA ***
 2910 INPUT "Valeur de la variable u = "
;U
 2920 IF U = INT(U) AND U<=0 THEN PRINT
"GAMMA(";U;") = infini ": END
 2940 REM *** Algorithme ****
 2950 LET P=1:LET U1=U
 2960 IF U<=5 THEN P=P*U: LET U=U+1: GOT
O2960
 2970 LET V=1/(U*U)
 2975 LET Y=(((.75*V-1)*V/3.5+1)*V/30-1)
*V/12+1
 2980 LET G=SQR(2*PI)*EXP((U-.5)*LN(U) -
U*Y)/P
```

```
2985 IF U1 = INT(U1) THEN G = INT(G+.5)
2990 PRINT"GAMMA(";U1;") = ";G
2995 END
```

EXEMPLES NUMÉRIQUES

A l'aide des quatre programmes nous avons dressé un petit tableau :

$J_n(x)$

n	x 1	2	3	4	5	6	7	8	9	Γ (n + 1)
0	0.77	0.22	-0.26	-0.4	-0.18	0.15	0.3	0.17	-0.09	1
1	0.44	0.58	0.34	-0.07	-0.33	-0.28	-.005	0.23	0.25	1
2	0.11	0.35	0.49	0.36	0.05	-0.24	-0.30	-0.11	0.14	2
3	0.02	0.13	0.31	0.43	0.36	0.11	-0.17	-0.29	-0.18	6
4	0	0.03	0.13	0.28	0.39	0.36	0.16	-0.11	-0.27	24
5	0	0	0.04	0.13	0.26	0.36	0.35	0.19	-0.06	120
6	0	0	0.01	0.05	0.13	0.25	0.34	0.34	0.2	720
7	0	0	0	0.02	0.05	0.13	0.23	0.32	0.33	5040
8	0	0	0	0	0.02	0.06	0.13	0.22	0.31	40320
9	0	0	0	0	0	0.02	0.06	0.13	0.21	362880

```
GAMMA(10 ) = 362880
GAMMA(.5 ) = 1.77245385
GAMMA(-2.5 ) = -.945308722
```

```
        Valeurs de la fonction d'erreur
I(SQR(2/PI)*EXP(-t^2/2),0, .2 ) =  .158519419
I(SQR(2/PI)*EXP(-t^2/2),0, .4 ) =  .310843483
I(SQR(2/PI)*EXP(-t^2/2),0, .6 ) =  .451493765
I(SQR(2/PI)*EXP(-t^2/2),0, .8 ) =  .576289203
I(SQR(2/PI)*EXP(-t^2/2),0, 1 ) = .682689492

I(SQR(2/PI)*EXP(-t^2/2),0, 1.2 ) =  .76986066
I(SQR(2/PI)*EXP(-t^2/2),0, 1.4 ) =  .838486682
I(SQR(2/PI)*EXP(-t^2/2),0, 1.6 ) =  .890401417
I(SQR(2/PI)*EXP(-t^2/2),0, 1.8 ) =  .928139363
I(SQR(2/PI)*EXP(-t^2/2),0, 2 ) = .954499737

I(SQR(2/PI)*EXP(-t^2/2),0, 2.5 ) =  .98758067
I(SQR(2/PI)*EXP(-t^2/2),0, 3 ) = .997300205
I(SQR(2/PI)*EXP(-t^2/2),0, 4 ) = .999936991
I(SQR(2/PI)*EXP(-t^2/2),0, 5 ) = .999999794
I(SQR(2/PI)*EXP(-t^2/2),0, 10 ) =  1
```

6. DÉRIVÉE NUMÉRIQUE ET CHAÎNAGE DE PROGRAMMES

INITIATION AU PROBLÈME

Nous avons déjà introduit les fonctions de Bessel de seconde espèce $Y_n(x)$ et proposé un programme pour calculer leurs valeurs si la variable x n'est pas de faible valeur. Ce dernier cas sera traité maintenant.

Notons d'abord la relation :

$$Y_n(x) = \frac{J_n(x)\cos n\pi - J_{o-n}(x)}{\sin n\pi}, \qquad n \neq \text{int}(n)$$

Nous trouvons donc facilement $Y_n(x)$ à l'aide des fonctions de Bessel de première espèce (voir Fonctions de Bessel-programme 1) si n n'est pas entier. Sinon, on applique la règle de l'Hôpital et on obtient :

$$Y_m(x) = \frac{1}{\pi}\frac{\partial J_n(x)}{\partial n}\Big|_{n=m} + \frac{(-1)^m}{\pi}\frac{\partial J_n(x)}{\partial n}\Big|_{n=-m}, \quad m = 0, \pm 1, \pm 2, \ldots$$

Pour ne pas trop compliquer la discussion, nous posons $m = 0$:

$$Y_0(x) = \frac{2}{\pi}\frac{\partial J_n(x)}{\partial n}\Big|_{n=0}$$

Notre premier programme permet alors le calcul de la dérivée numérique d'une fonction et, par un chaînage de ce programme aux programmes « Fonctions de Bessel » et « Fonction Gamma », on obtient $Y_0(x)$.

Remarques

1. Rappelons la notation de Landau; on dit :

$$f(h)=O(g(h))$$

si le rapport $\frac{f}{g}$ reste borné par une constante au voisinage de $h=0$.

2. L'ordre O de la méthode correspond au nombre d'itérations.

MÉTHODE

Expliquons d'abord le programme « Dérivée numérique d'après Romberg ».

Nous notons les développements de Taylor de $f(x+h)$ et $f(x-h)$:

$$f(x\pm h)=f(x)\pm hf'(x)+\frac{h^2}{2}f''(x)\pm\frac{h^3}{6}f^{(3)}(x)+\frac{h^4}{24}f^{(4)}(x)+O(h^5)$$

D'où l'on obtient les différences :

$$D(k)=\frac{f(x+h_k)-f(x-h_k)}{2\,h_k},\quad h_k=\frac{h}{2^{k-1}},\quad k=1,2,\ldots$$

$$=f'(x)+\frac{h_k^2 f^{(3)}(x)}{6}+O(h_k^4)$$

et, par une première extrapolation :

$$f'(x)=D(k+1)+\frac{D(k+1)-D(k)}{4-1}+O(h^4)$$

En réitérant ce procédé, nous trouvons le schéma suivant :

$$D(1)=\frac{f(x+h)-f(x-h)}{2\,h}$$

$$D(2)=\frac{f\left(x+\frac{h}{2}\right)-f\left(x-\frac{h}{2}\right)}{h} \qquad D(1) \longleftarrow D(2)+\frac{D(2)-D(1)}{3}$$

$$D(3)=\frac{f\left(x+\frac{h}{4}\right)-f\left(x-\frac{h}{4}\right)}{\frac{h}{2}} \qquad D(2) \longleftarrow D(3)+\frac{D(3)-D(2)}{3}$$

$$D(1) \longleftarrow D(2)+\frac{D(2)-D(1)}{15}$$

$$=f'(x)+O(h^6)$$

etc. (selon l'ordre de la méthode).

Nous continuons jusqu'à ce que deux approchées successives D(1) ne diffèrent plus (précision PR).

UTILISATION DES PROGRAMMES

a. Dérivée numérique

Il faut introduire :

- la valeur de la variable x : X = ?
- l'ordre de la méthode o : O = ?
- la précision PR : PR = ?
- l'intervalle de calcul h : H = ?

Pour réduire l'influence des erreurs d'arrondi, on choisit :

$$0.2 \leqq h \leqq 0.5 \quad \text{et} \quad 2 \leqq o \leqq 7$$

ou :

$$h < 0.2 \quad \text{et} \quad 1 \leqq o \leqq 3$$

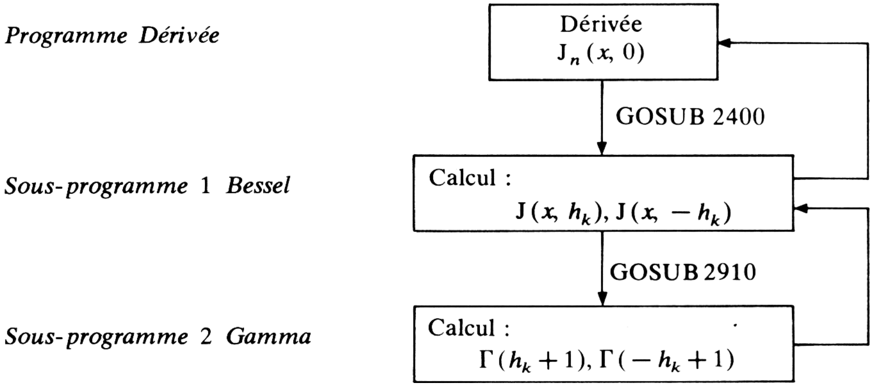
b. Programme « Chaînage »

Se basant sur la définition :

$$Y_0(x) = \frac{2}{\pi} \lim_{n \to 0} \frac{J_n(x) - J_0(x)}{n}$$

on introduit la valeur de la variable x et on calcule :

$$\frac{dJ_n(x, n)}{dn}\bigg|_{n=0} = :J_n(x, 0)$$



LES PROGRAMMES

```
10 REM *** Derivee numerique ***
20 REM *** d'apres ROMBERG
90 REM *** Fonction f(x)      ***
100 DEF FN F(X) = ATN(X)
110 PRINT " f(x) = arctan x "
190 REM *** Valeur de la variable x ***
200 INPUT " f'(x)- variable x = ";X
210 INPUT " Ordre de la methode o= ";O
220 IF (O<=0) OR (O<>INT(O)) THEN PING:
GOTO 210
```

```
230 INPUT " Precision PR = ";PR
240 INPUT " Intervalle de calcul h= ";H
250 IF (H<=0) OR (H>.5) THEN PING: GOTO
240
260 DIM D(O+1)
280 PRINT " Tableau des approchees :"
290 REM *** Algorithme ***
300 DEF FN D(H) = (FN F(X+H) - FN F(X-H
))/(2*H)
310 D(1) = FN D(H): D1=D(1)
320 PRINT " K = 0 :",D1
330 FOR K =1 TO O
340 H=H/2
350 PRINT " K = ";K
360 D(K+1) = FN D(H)
370 R=1: REM *** Extrapolation ***
380 FOR K1 = K TO 1 STEP -1
390 R=4*R
400 D(K1) =D(K1+1)+ (D(K1+1) - D(K1))/(
R-1)
410 PRINT D(K1),
420 NEXT K1
430 D2 = D(1)
440 IF ABS(D1 - D2)<= ABS(PR) THEN PRIN
T:PRINT " Derivee f'(";X;") = ";D2: END
450 D1 = D2
460 PRINT: NEXT K
470 PRINT " Precision ";PR;" non obtenu
e, augmentez o "
480 PRINT " Derniere approchee : "
490 PRINT " f'(";X;") = ";D2: END
```

```
5 REM *** Chainage              ***
10 REM *** Derivee numerique ***
20 REM *** d'apres ROMBERG   ***
110 PRINT "f(n)= J(x,n)"
115 INPUT "Valeur de la variable x = ";
X
120 IF (X<=0) OR (X>=7+ABS(N)) THEN PRI
NT"x >= 7 + abs(n); voir programme 2"
```

```
190 REM *** Valeur de la variable x ***
210 INPUT " Ordre de la methode o= ";O
230 INPUT " Precision PR = ";PR
240 INPUT " Intervalle de calcul h= ";H
260 DIM D(O+1)
280 PRINT " Tableau des approchees :"
290 REM *** Algorithme ***
300 N=H:GOSUB 2400:J1=J:N=-H:GOSUB2400
310 D(1) =(J1-J)/(2*H): D1=D(1)
320 PRINT " K = 0 :",D1
330 FOR K =1 TO O
340 H=H/2
350 PRINT " K = ";K
355 N=H:GOSUB2400:J1=J:N=-H:GOSUB2400
360 D(K+1) = (J1-J)/(2*H)
370 R=1: REM *** Extrapolation ***
380 FOR K1 = K TO 1 STEP -1
390 R=4*R
400 D(K1) =D(K1+1)+ (D(K1+1) - D(K1))/(
R-1)
410 PRINT D(K1),
420 NEXT K1
430 D2 = D(1)
440 IF ABS(D1 - D2)<= ABS(PR) THEN PRIN
T:PRINT" Derivee Jn(";X;",0) = ";D2:GOTO
500
450 D1 = D2
460 PRINT: NEXT K
470 PRINT " Precision ";PR;" non obtenu
e, augmentez o "
480 PRINT " Derniere approchee : "
500 PRINT "Y0(";X;") = ";2*D2/PI: END
2400 REM *** Fonctions de BESSEL ***
2405 REM *** J(x,n), Programme 1 ***
2430 REM *** Voir Programme fonction  g
amma ***
2450 GOSUB 2910
2470 REM *** Algorithme ***
2480 B=1: M=1: J=B
2490 B = -B*X*X/(4*M*(N+M))
2500 J = J + B: M=M+1
2510 IF ABS(B)>1E-9  THEN 2490
```

```
2520 J=J*(X/2)^N/G
2530 RETURN
2900 REM *** Fonction gamma(N+1) ***
2910 U=N+1
2950 LET P=1:LET U1=U
2960 IF U<=5 THEN P=P*U: U=U+1: GOTO296
0
2970 LET V=1/(U*U)
2980 LET Y=(((.75*V-1)*V/3.5+1)*V/30-1)
*V/12+1
2985 G=SQR(2*PI)*EXP((U-.5)*LN(U)-U*Y)/
P
2990 RETURN
```

EXEMPLES NUMÉRIQUES "DÉRIVÉES"

Exemple 1

```
f(x) = arctan x
f'(x)- variable x = ? 1
Ordre de la methode o= ? 5
Precision PR = ? 1E-08
Intervalle de calcul h= ? .5
Tableau des approchees :
K = 0 :           .519146114
K =  1
.500429363

K =  2
.500025076  .499998123

K =  3
.500001535  .499999966  .499999995

K =  4
.500000092  .499999996  .499999996
.499999996

Derivee f'( 1 ) =  .5
```

Exemple 2

```
f(x) = ln(x+SQR(x^2-1))
f'(x)- variable x = ? 2
Ordre de la methode o= ? 4
Precision PR = ? 1E-06
Intervalle de calcul h= ? .5
Tableau des approchees :
K = 0 :             .604375588
K =  1
.576579214

K =  2
.577308897  .577357542

K =  3
.577347771  .577350362  .577350248

K =  4
.577350114  .57735027  .577350268
.577350268

Derivee f'( 2 ) =  .57735


valeur exacte: .57735027
```

EXEMPLES NUMÉRIQUES "CHAÎNAGES"

Nous allons calculer quelques valeurs de la fonction de Bessel de seconde espèce $Y_0(x)$:

```
f(n)= J(x,n)
Valeur de la variable x = 1
 Ordre de la methode o= 4
 Precision PR = 1E-08
 Intervalle de calcul h= .2
```

```
 Tableau des approchees :
 K = 0 :    .156077452
 K = 1
.138689566
 K = 2
.138637218          .138633728
 K = 3
.138633955          .138633738          .138633738
Derivee Jn(1 ,0) = .138633738
Y0(1 ) = .08825698
valeur theorique: .0882569642

f(n)= J(x,n)
Valeur de la variable x = .1
 Ordre de la methode o= 6
 Precision PR = 2E-07
 Intervalle de calcul h= .1
 Tableau des approchees :
 K = 0 :   -2.40930404
 K = 1
-2.41003003
 K = 2
-2.40997976          -2.40997641
 K = 3
-2.40997675          -2.40997655          -2.40997655
Derivee Jn(.1 ,0) = -2.40997655
Y0(.1 ) = -1.5342388
valeur theorique: -1.534238651
```

7. INTÉGRATION

INITIATION AU PROBLÈME

Partons de l'équation différentielle d'un objet en rotation obéissant à la loi de la pesanteur (voir la Figure 7.1):

$$mL^2\ddot{z} + Lmg \sin z = 0$$

- mL^2 moment d'inertie
- $z(t)$ angle
- $Lmg \sin z$ couple de rotation
- t temps.

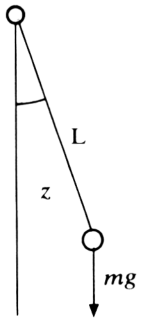


Figure 7.1.

Nous trouvons facilement l'intégrale première de l'énergie cinétique en intégrant :

$$\dot{z}^2 = \frac{2g}{L} \cos z + C$$

Admettons les conditions initiales suivantes :

$$z(0) = z_0, \quad \dot{z}(0) = 0, \quad -\pi < z_0 < 0$$

L'objet va donc descendre et il en résulte, pour t de faible valeur :

$$\dot{z}^2 = \frac{2g(\cos z - \cos z_0)}{L}$$

$$t(z) = \sqrt{\frac{L}{2g}} \int_{z_0}^{z} \frac{dy}{\sqrt{\cos y - \cos z_0}}$$

$$= \sqrt{\frac{L}{g}} \int_{z_0}^{z} \frac{d\left(\frac{y}{2}\right)}{\sqrt{\sin^2\left(\frac{z_0}{2}\right) - \sin^2\left(\frac{y}{2}\right)}} \tag{7.1}$$

On ne trouve pas de primitive pour évaluer cette intégrale. De plus, la fonction à intégrer présente des singularités à $y = \pm z_0$.

Nous voulons d'abord écarter ce problème supplémentaire et étudier une méthode pour calculer l'intégrale définie :

$$I = \int_a^b f(x)\, dx$$

d'une fonction régulière (continue et bornée) sur l'intervalle d'intégration $[a, b]$, $-\infty < a < b < \infty$.

Cela nous donnera l'écart de temps :

$$Dt = t(z_2) - t(z_1), \qquad z_0 < z_1 < z_2 < -z_0$$

que l'objet met à passer d'une position z_1 à l'autre.

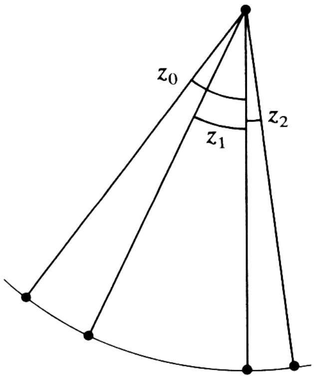


Figure 7.2.

MÉTHODE

Nous choisissons la méthode de Romberg et gagnons une première valeur approchée à l'aide de la méthode du trapèze :

$$I(1)=\frac{f(a)+f(b)}{2}\,h, \qquad h=b-a$$

Ensuite nous partageons l'intervalle $[a, b]$ d'une façon régulière :

$$h \longleftarrow \frac{h}{2}$$

et nous trouvons de même :

$$I(2)=\left(f(a)+2\,f\left(\frac{a+b}{2}\right)+f(b)\right)\frac{b-a}{4}$$

En combinant :

$$I(1) \longleftarrow I(2)+\frac{I(2)-I(1)}{4-1}$$

on obtient la formule de Simpson bien connue et souvent appliquée :

$$I \approx \frac{b-a}{6}\left(f(a)+4\,f\left(\frac{a+b}{2}\right)+f(b)\right)$$

Procédé général :

Arrivés à $h=\frac{(b-a)}{2^k}$, $k=1, 2, ..., n$ (n : ordre de la méthode), nous posons :

$$I(k+1)=\frac{h}{2}\left(f(a)+f(b)+\sum_{i=1}^{2^k-1} f(a+ih)\right)$$

$$=\frac{1}{2}\,I(k)+h\sum_{i=0}^{2^{k-1}-1} f(a+(2\,i+1)\,h)$$

et ensuite pour $k1 = k,\ k-1,\ \ldots,\ 1$:

$$I(k1) \longleftarrow I(k1+1) + \frac{(I(k1+1) - I(k1))}{(4^{k-k1+1} - 1)}$$

$I(1)$ nous livre une meilleure approximation de l'intégrale I. L'erreur $I - I(1)$ aura la forme :

$$I - I(1) = (-1)^{n+1} \frac{(b-a)^{2n+3}}{2^{n(n+1)}} \frac{B_{2n+2}}{(2n+2)!} f^{(2n+2)}(C)$$

(B_i nombre de Bernoulli, $C \in [a, b]$.)

ORGANIGRAMME

Intégration d'après Romberg

Entrées	Fonction $f(x)$.
100-285	Intervalle d'intégration $[a, b]$, $L = b - a$.
	Ordre n de la méthode.
	Précision relative P.

Initialisation $\quad I(1) \longleftarrow \dfrac{L(f(a)+f(b))}{2}$

300-320 $\quad I1 \longleftarrow I(1), \quad M \longleftarrow 2$

$k = 1, \ldots, n$: $h \longleftarrow \dfrac{L}{M}$, $S \longleftarrow 0$, $R \longleftarrow 1$

a. Calcul $I(k+1)$:

$k1 = 1, 3, 5, \ldots, M-1$

$S \longleftarrow S + f(a + k1\,h)$

$$I(k+1) = \frac{I(k)}{2} + Sh$$

b. Calcul d'une meilleure approchée I(1) :

$k1 = k, k-1, \ldots, 1$

$R \longleftarrow 4R$

$$I(k1) = I(k1+1) + \frac{(I(k1+1) - I(k1))}{(R-1)}$$

$I2 \longleftarrow I(1)$, ◇ $I2 = 0?$ — OUI ⟶ FIN

↓ NON

c. Précision relative P obtenue?

⬡ $|I2 - I1| \leqq P * |I2|?$ — OUI ⟶ $I = I2$ ⟶ FIN

⟶ NON $M \longleftarrow 2M$, $I1 \longleftarrow I2$

◇ $k = n?$ — OUI ⟶ précision non obtenue ⟶ FIN

↓ NON

$k \longleftarrow k+1$ (retour en a.)

Variables

A, B	Limites (inférieure et supérieure) de l'intervalle.
L	$L = B - A$.
N	Ordre de la méthode.
P	Précision relative.
R	Variable d'incrémentation.
K, K1	Variables de boucles.
I(K), I(K1), I1, I2	Valeurs approchées de l'intégrale I.
M, H	Subdivision de l'intervalle, $M = 2^k$, $H = \frac{L}{M}$.
S	Somme.

PROGRAMME

```
 10 REM *** Integration ***
 20 REM *** d'apres ROMBERG ***
 90 REM.*** Fonction f(x) ***
 100 DEF FN F(X) = X^7
 110 PRINT "f(x) = x^7"
 190 REM *** Intervalle [a,b] ***
 200 INPUT "Limite inferieure a = ";A
 210 INPUT "Limite superieure b = ";B
 220 IF B<=A THEN PING: GOTO 200
 230 PRINT "[";A;",";B;"] Intervalle "
 240 L=B-A
 250 INPUT "Nombre d'extrapolations: ord
re de la  methode n = ";N
 260 IF (N<=0) OR (N<>INT(N)) THEN PING
: GOTO 250
 270 PRINT "Precision relative = ";
 280 INPUT P: REM *** P>=1E-9 ! ***
 290 PRINT "Tableau des valeurs approche
es:":PRINT
 295 REM *** Algorithme ***
 300 DIM I(N+1)
 310 I(1) = L*(FN F(A) + FN F(B))/2
 320 M=2: REM *** Subdivision de l'inter
valle ***
 330 FOR K=1 TO N
 340 H = L/M: S=0 : I1=I(1):PRINT"K = ";
K
 350 FOR K1 = 1 TO M-1 STEP 2
 360 S=S + FN F(A+K1*H)
 370 NEXT K1
 380 I(K+1) = I(K)/2 + S*H
 390 R=1 : REM *** Extrapolation ***
 400 FOR K1 = K TO 1 STEP -1
 410 R = 4*R
 420 I(K1) = I(K1+1)+(I(K1+1)-I(K1))/(R-
1)
 430 PRINT I(K1),
 440 NEXT K1
```

```
 450 PRINT: I2=I(1):IF ABS(I2)<P THEN 52
0
 460 IF ABS(I1-I2) <= ABS(P*I2) THEN PRI
NT "Integrale I(f,a,b)= ";I2: END
 470 M=2*M
 480 NEXT K
 490 PRINT "Precision relative ";P;" non
 obtenue, augmentez la valeur de N."
 500 PRINT "Dernieres valeurs approchees
: "
 510 PRINT "I1 = ";I1;" I2 = ";I2
 515 PRINT"|(I1 - I2)/I2| = ";ABS((I1-I2
)/I2): END
 520 PRINT "I = 0 ? Changez la limite su
perieure b ": END
```

DÉROULEMENT DU PROGRAMME

Le programme calcule une valeur approchée I(1) = I2 de l'intégrale :

$$I = \int_a^b f(x)\, dx$$

f étant une fonction continue, bornée, ou f donnée en forme de tableau :

$$f_k = f\left(\frac{a + k(b - a)}{2^N}\right), \qquad k = 0, \ldots, 2^N$$

$[a, b]$ désigne un intervalle borné.

On introduit en lignes :

100 à 110	Fonction f : DEF FN F(X) = ...
200 à 210	A : limite inférieure; B : limite supérieure de l'intervalle.
250	Ordre N de la méthode.
270	Précision relative P.

Si la précision n'est pas obtenue, le programme affiche les deux dernières approchées I1 et I2 ainsi que l'erreur relative :

$$abs\left(\frac{(I1-I2)}{I2}\right)$$

Dans ce cas-là, il faut augmenter N ou appliquer le programme plusieurs fois après un partage de l'intervalle (intégration par arcs).

Si, au cours du calcul, I(1) devient nulle, le programme s'arrête, la notion de précision relative n'est plus appropriée et il faut subdiviser l'intervalle afin d'arriver à deux intégrales différentes de zéro.

EXEMPLES NUMÉRIQUES

```
f(x) = x^7
Limite inferieure a = ? 0
Limite superieure b = ? 1
[ 0 , 1 ] Intervalle
Nombre d'extrapolations: ordre de la  me
thode n = ? 3
Precision relative = ? 1E-08
Tableau des valeurs approchees:

K =  1
 .171875
K =  2
 .129150391        .126302083
K =  3
 .125278473        .125020345        .125

Precision relative  1E-08  non obtenue,
augmentez la valeur de N.
Dernieres valeurs approchees:
I1 =  .126302083  I2 =  .125
!(I1 - I2)/I2! =  .010416666
```

```
f(x) = x^7
Limite inferieure a = ? 0
Limite superieure b = ? 1
[ 0 , 1 ] Intervalle
Nombre d'extrapolations: ordre de la  me
thode n = ? 4
Precision relative = ? 1E-08
Tableau des valeurs approchees:

K =  1
 .171875
K =  2
 .129150391        .126302083
K =  3
 .125278473        .125020345        .125

K =  4
 .125017703        .125000318        .125
 .125
Integrale I(f,a,b)=  .125
```

Le résultat est exact pour $n \geqq 3$, parce que $\dfrac{d^8 f}{dx^8} = 0$.

```
f(x) = 1/SQR(.25sin^2(x)+cos^2(x))
Limite inferieure a = ? 0
Limite superieure b = ? 1.57079633
[ 0 , 1.57079633 ] Intervalle
Nombre d'extrapolations: ordre de la  me
thode n = ? 5
Precision relative = ? 1E-09
Tableau des valeurs approchees:

K =  1
 2.11000994
K =  2
 2.1516789         2.15445683
K =  3
 2.15647149        2.156791
 2.15682805
```

```
K =  4
 2.15651565       2.15651859
 2.15651427       2.15651304
K =  5
 2.15651566       2.15651566
 2.15651561       2.15651561
 2.15651562
Precision relative  1E-09  non obtenue,
augmentez la valeur de N.
Dernieres valeurs approchees:
I1 =  2.15651304  I2 =  2.15651562
|(I1 - I2)/I2| =  1.19496912E-06


f(x) = 1/SQR(.25sin^2(x)+cos^2(x))
Limite inferieure a = ? 00
Limite superieure b = ? 1.57079633
[ 0 , 1.57079633 ] Intervalle
Nombre d'extrapolations: ordre de la  me
thode n = ? 8
Precision relative = ? 1E-09
Tableau des valeurs approchees:

K =  1
 2.11000994
K =  2
 2.1516789        2.15445683
K =  3
 2.15647149       2.156791
 2.15682805
K =  4
 2.15651565       2.15651859
 2.15651427       2.15651304
K =  5
 2.15651566       2.15651566
 2.15651561       2.15651561
 2.15651562
K =  6
 2.15651565       2.15651565
 2.15651565       2.15651565
 2.15651565       2.15651565
```

```
K =  7
 2.15651565        2.15651565
 2.15651565        2.15651565
 2.15651565        2.15651565
 2.15651565
Integrale I(f,a,b)=  2.15651565

 Valeur exacte I = 2.1565156475... .
```

```
f(x) = sin(10*Pi*x)
Limite inferieure a = ? 0
Limite superieure b = ? 1.5
[ 0 , 1.5 ] Intervalle
Nombre d'extrapolations: ordre de la  me
thode n = ? 10
Precision relative = ? 1E-08
Tableau des valeurs approchees:

K =  1
-1
K =  2
-.957106781      -.954247233
K =  3
-.955058177      -.954921603
-.954932308
K =  4
 .326520142       .411958696
 .433655209       .43910065
K =  5
 .0658802655      .0485042738
 .042735156       .0412021361
 .0408131835
K =  6
 .0637730468      .0636325655
 .0638726971      .0639555894
 .0639778313      .0639834881
```

```
K =  7
 .0636685797      .0636616152
 .0636620763      .0636612504
 .0636609627      .0636608853
 .0636608656
K =  8
 .0636623839      .0636619708
 .0636619765      .0636619761
 .0636619768      .0636619771
 .0636619771      .0636619772
K =  9
 .063662003       .0636619776
 .0636619777      .0636619777
 .0636619777      .0636619777
 .0636619777      .0636619777
 .0636619777
Integrale I(f,a,b)=  .0636619777
```

```
f(x) = sin(10*Pi*x)
Limite inferieure a = ? 1.5
Limite superieure b = ? 2
[ 1.5 , 2 ] Intervalle
Nombre d'extrapolations: ordre de la  me
thode n = ? 10
Precision relative = ? 1E-08
Tableau des valeurs approchees:

K =  1
-.333333333
K =  2
 .15236893        .184749081
K =  3
-.072940449      -.0879610743
-.0922898069
K =  4
-.0640324614     -.0634385955
-.0630493498     -.0629346814
```

```
K =  5
-.0636831167     -.063659827
-.0636633387     -.0636657465
-.0636664611
K =  6
-.06366327       -.0636619469
-.0636619806     -.0636619752
-.0636619715     -.0636619704
K =  7
-.0636620578     -.063661977
-.0636619774     -.0636619774
-.0636619774     -.0636619774
-.0636619774
K =  8
-.0636619823     -.0636619772
-.0636619772     -.0636619772
-.0636619772     -.0636619772
-.0636619772     -.0636619772
Integrale I(f,a,b)= -.0636619772
```

GÉNÉRALISATIONS

Nous reprenons la formule (7.1) et nous posons :

$$u=\sin\left(\frac{y}{2}\right), \quad v=\sin\left(\frac{z}{2}\right), \quad k=\sin\left(\frac{z_0}{2}\right)$$

D'où :

$$t=\sqrt{\frac{L}{g}}\int_k^v \frac{1}{\sqrt{-k+u}}\,\frac{1}{\sqrt{-k-u}\,\sqrt{1-u^2}}\,du \qquad (7.2)$$

Nous sommes conduits à étudier le calcul d'intégrales :

$$\int_a^b s(x)f(x)\,dx, \quad s=\frac{1}{\sqrt{x-k}}, \quad f=\frac{1}{\sqrt{(x^2-1)(k+x)}}$$

où s présente une singularité et où la fonction f est régulière.

On peut opérer de plusieurs façons :

1. On pose :

$$a_j = a + \frac{b-a}{M_j}, \quad M_j \uparrow \infty, \quad M_1 = 1$$

(La suite (M_j) est donc strictement croissante vers l'infini.)

On calcule :

$$I_j := \int_{a_j}^{a_{j+1}} s(x)\, f(x)\, dx$$

et on trouve $I = I_1 + I_2 + \ldots + \ldots$

2. Une ou plusieurs intégrations par parties « améliorent » les fonctions à intégrer.

3. On remplace f par un polynôme approprié.

Exemple

Reprenons l'exemple du premier paragraphe. Soit :

$$z_0 = \frac{-\pi}{2}, \quad z_1 = \frac{-\pi}{4}, \quad z_2 = \frac{\pi}{4} \quad \text{et} \quad \sqrt{\frac{L}{g}} = 1\ s$$

Nous trouvons pour l'écart de temps que le pendule met à passer de l'angle $\frac{-\pi}{4}$ à $\frac{\pi}{4}$:

$$Dt = \int_{-\pi/4}^{\pi/4} \frac{d\left(\frac{y}{2}\right)}{\sqrt{0{,}5 - \sin^2\left(\frac{y}{2}\right)}}\ s$$

$$= \int_{-\pi/4}^{\pi/4} \frac{dy}{\sqrt{2 - 4 \sin^2\left(\frac{y}{2}\right)}}\, s = 1{,}1750647\ s$$

Discutons encore l'intégrale généralisée :

$$I = \lim_{z \to -\pi/2} \int_{z}^{0} \frac{dx}{\sqrt{2 - 4 \sin^2\left(\frac{x}{2}\right)}}$$

et appliquons la première méthode proposée dans ce paragraphe. Nous trouvons ainsi :

, Intégrale I(*f*, − 1,47079633, 0) = 1,40679
, Intégrale I(*f*, − 1,56079633, − 1,47079633) = 0,30587
, Intégrale I(*f*, − 1,56979633, − 1,56079633) = 0,0967
, Intégrale I(*f*, − 1,57069633, − 1,56979633) = 0,03058
, Intégrale I(*f*, − 1,57078633, − 1,57069633) = 9,67 E − 03
, Intégrale I(*f*, − 1,57079533, − 1,57078633) = 3,06 E − 03
, Intégrale I(*f*, − 1,57079623, − 1,57079533) = 9,7 E − 04

I = 1,8537

La seconde méthode nous fournit après une intégration de (7.2) par parties :

```
 99  Z = − 1/SQR(2)
100  DEF FN F(X) = SQR(X − Z)*(3*X^2 + 2*Σ*X − 1)/((X^2 − 1)
     *(X + Z))^1,5
```

I = 1,8541

III
RÉSOLUTION D'ÉQUATIONS; POLYNÔMES

8. RÉSOLUTION D'UNE ÉQUATION NON LINÉAIRE

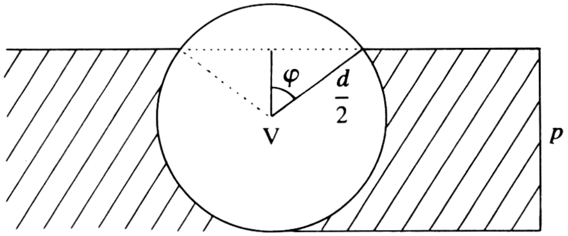
INITIATION AU PROBLÈME

Considérons un tronc cylindrique d'un poids spécifique de 7 N/dm^3 d'une longueur ℓ et de diamètre d plongé dans l'eau. Jusqu'à quelle profondeur va-t-il disparaître sous l'eau (poids spécifique 9,80665 N/dm^3)?

Il faut donc trouver le volume partiel V sous l'eau du tronc (voir ci-dessous) qui, si l'on observe la figure, s'avère être :

$$V \mathrel{\widehat{=}} \text{secteur circulaire} + 2 \text{ triangles}$$

$$V = \ell \left(\frac{d}{2}\right)^2 (\pi - \varphi + \sin\varphi \cos\varphi)$$



La loi d'Archimède nous livre l'équation à résoudre :

$$9{,}80665\,(\pi - \varphi + \sin\varphi\cos\varphi) = 7\,\pi \iff \varphi = \frac{1}{2}\sin 2\varphi + 0{,}89912$$

ayant éliminé le facteur commun $\ell \left(\frac{d}{2}\right)^2$.

L'équation équivaut ($\cos = x$) à :

$$\arccos x - x\sqrt{1-x^2} = 0{,}89912, \qquad p = (1+x)\,d$$

dont il faut calculer la solution x, $0 < x < 1$.

MÉTHODE

Il s'agit de résoudre une équation non linéaire $f(x) = 0$. Nous choisissons la méthode itérative de King et Van de Vel qui améliore la méthode bien connue de Newton :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

en cas de racines multiples où le rapport $\dfrac{f(x)}{f'(x)}$ risque de prendre des valeurs aléatoires [1].

Posons :

$$u_n = \frac{f(x_n)}{f'(x_n)}$$

et partons d'un groupe de valeurs initiales :

$$x_0,\ m_0,\ x_1 = x_0 - m_0\, u_0$$

La relation itérative s'écrit donc :

$$m_{n+1} = m_n \frac{u_n}{u_n - u_{n+1}} \qquad x_{n+2} = x_{n+1} - m_{n+1}\, u_{n+1}$$

1. R. F. King, *Improving the Van de Vel root finding method*. Computing 30 (1983), 373-378.

L'ordre de convergence s'avère être : $\frac{1+\sqrt{5}}{2}$, lim m_n la multiplicité de la formule en question.

Plus l'ordre est grand, plus la convergence est rapide.
Le programme demande la définition de la fonction f et de la dérivée f' :

$$30 \text{ DEF FN } U(X) = \left(\frac{f(x)}{f'(x)}\right)$$

Ensuite on introduit les valeurs initiales x_0, m_0 et la précision envisagée.

PROGRAMME

```
 10 REM *** Resolution d'une equation  n
on-lineaire f(x) = 0 ***
 20 REM *** Introduction de la fonction
u = f/f' ***
 25 DEF FNF(X)=((((324*X-351)*X+108)*X+6
)*X-8)*X+1
 29 DEF FN G(X)=(((1620*X-1404)*X+324)*X
+12)*X-8
 30 DEF FN U(X)=FN F(X)/ FN G(X)
 40 REM *** Valeurs initiales x0, m0 ***
 50 PRINT " Valeurs initiales x0, m0 = "
;:INPUT X0,M0:PRINT X0;",";M0
 60 PRINT " Precision P = ";:INPUTP:PRIN
T P
 90 REM *** Algorithme de VAN DE VEL -
KING ***
 100 X1 = X0 - M0*FN U(X0)
 110 PRINT" 1",X1,M0
 120 N=0
 125 M1=M0
 130 M0 = M0*FN U(X0)/(FN U(X0) - FN U(X
1))
```

```
 140 X0 = X1
 150 X1 = X1 - M0*FN U(X1)
 160 IF ABS(X1 - X0) < P THEN PRINT" Sol
ution x = "; X1: END
 170 PRINT N+2, X1,M0:N=N+1
 180 IF N > 20 THEN PRINT " Precision no
n obtenue apres ";N;"  iterations.":END
 190 GOTO 125
```

EXEMPLES NUMÉRIQUES

Tronc cylindrique

```
 10 REM *** Resolution d'une equation  n
on-lineaire f(x) = 0 ***
 20 REM *** Introduction de la fonction
u = f/f' ***
 25 PRINT"f(x)=arccos x - x*SQR(1-x^2)-.
89912"
 30 DEF FN U(X)=ATN(SQR(1-X*X)/X)/(-2*SQ
R(1-X*X))+X/2+.44956/SQR(1-X*X)
```

```
f(x)=arccos x - x*SQR(1-x^2)-.89912
 Valeurs initiales x0, m0 = .5 ,.1
 Precision p = 1E-08
1  .335492614
2  .342358736
3  .342668025
4  .342667602
 Solution x = .342667602
```

```
f(x)=arccos x - x*SQR(1-x^2)-.89912
 Valeur  initiale  x0 = .5
 Precision p = 1E-08
 Algorithme de NEWTON
1  .335492614
2  .342657793
3  .342667602
 Solution x = .342667602
```

```
f(x)=324x^5-351x^4+108x^3+6x^2-8x+1
 Valeurs initiales x0, m0 = 1 ,1
 Precision P = 1E-04
1  .852941176
2  .298856984
3  .337681141
4  .333365427
5  .337662319
 Solution  x = .3376
```

```
 f(x)=324x^5-351x^4+108x^3+6x^2-8x+1
 Valeurs initiales x0, m0 = 1 ,1
 Precision P = 1E-05
1  .852941176
2  .298856984
3  .337681141
4  .333365427
5  .337662319
6  .337643844
7  .345784782
8  .333272965
9  .346102675
10           .346437053
11           .333325114
12           .346103966
13           .345787314
14           .33339066
 Solution  x = .33339
```

```
 x=1/3:  zero de l'ordre 4
```

```
 Valeurs initiales x0, m0 = -.3 ,1
 Precision P = 1E-06
1  -.262
2  -.246310395
3  -.250299529
4  -.250007623
5  -.24999984
 Solution  x = -.25
```

```
f(x)=324x^5-351x^4+108x^3+6x^2-8x+1
 Valeur initiale  x0 = 1
 Precision P = 1E-04
Algorithme de NEWTON
1 .852941176
2 .736726698
3 .645229793
4 .573503045
5 .517540875
6 .47409567
7 .440536061
8 .41473669
9 .394990381
10 .379935897
11 .368496814
12 .359829141
13 .353276433
14 .348331571
15 .344605594
16 .341801054
17 .339693641
18 .338109001
19 .336919298
20 .336011195
21 .335325783
22 .33489903
23 .334579745
24 .334103856
 Solution  x = .3341
```

9. POLYNÔME CARACTÉRISTIQUE, DÉTERMINANT (λI − A) D'UNE MATRICE A ET LA MATRICE COMPLÉMENTAIRE

INITIATION AU PROBLÈME [1]

Les systèmes différentiels linéaires à coefficients constants qu'on peut écrire sous la forme matricielle :

$$y'(x) = \mathrm{A}\,y(x) + b(x), \quad \mathrm{A}\,(n,\ n)\text{-matrice}$$

ont un grand intérêt pratique.

On les rencontre dans la plupart des problèmes de mouvements en physique quand on linéarise leurs équations différentielles de la forme :

$$z'' = \mathrm{F}\,(x,\ z,\ z')$$

après avoir posé :

$$\vec{y} = \begin{pmatrix} z \\ z' \end{pmatrix}$$

Cette équation est d'ailleurs une conséquence du principe fondamental de la dynamique :

$$\vec{\mathrm{F}} = m\vec{a}$$

$\vec{\mathrm{F}}$ désigne la résultante des forces agissant sur la masse m;

$\vec{a}$ désigne le vecteur accélération.

Prenons comme exemple le pendule pesant (voir le Chapitre 7) qui obéit à l'équation $\ddot{z} = -\left(\frac{g}{L}\right)\sin z$ (L étant sa longueur, g constante de gravitation universelle).

On introduit :

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} z \\ \dot{z} \end{pmatrix}$$

On trouve :

$$\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ -\frac{g}{L}\sin y_1 \end{pmatrix}$$

et, en remplaçant sin y_1 par y_1, on aboutit à :

$$\dot{\vec{y}} = \begin{pmatrix} 0 & 1 \\ -\frac{g}{L} & 0 \end{pmatrix} \vec{y}$$

L'intégration du système homogène $\vec{y}' = A\vec{y}$ ou :

$$y_i'(x) = \sum_{j=1}^{n} a_{ij}\, y_j(x),\ i = 1, \ldots, n \qquad (*)$$

se ramène à un problème purement algébrique :
en posant $\vec{y} = \vec{y_0}\, e^{\lambda x}$, $(*)$ équivaut à $(\lambda I - A)\vec{y_0} = \vec{0}$.

Ce système d'équations linéaires possède une solution non nulle si et seulement si λ est une racine du polynôme caractéristique :

$$P(\lambda) = \text{dét}(\lambda I - A) = \lambda^n - \text{tr } A\,\lambda^{n-1} + \ldots + (-1)^n \text{ dét } A$$

$$\text{tr } A := a_{11} + a_{22} + \ldots + a_{nn}$$

On a donc intérêt à connaître le polynôme caractéristique.

MÉTHODE

Nous appliquons la méthode de Fadeev (*) et nous posons :

$$P(\lambda) = \lambda^n - p_1 \lambda^{n-1} - p_2 \lambda^{n-2} - \dots - p_n$$

Ensuite, nous introduisons la matrice complémentaire $B(\lambda)$ de $\lambda I - A$, ce qui veut dire :

$B(\lambda)$ remplit la relation :

$$B(\lambda) \cdot (\lambda I - A) = P(\lambda) = \text{dét}(\lambda I - A).$$

Il en résulte :

$$B(\lambda) = B_0 \lambda^{n-1} + B_1 \lambda^{n-2} + \dots + B_{n-1}, \quad B_0 = I$$

et :

$$B_k = A^k - p_1 A^{k-1} - \dots - p_k I$$

où :

$$B_k = AB_{k-1} - p_k I, \quad k = 1, \dots, n-1$$

Si A est inversible, on a :

$$A^{-1} = \frac{B_{n-1}}{p_n}$$

$$p_n = (-1)^{n-1} \text{ dét } A$$

B_{n-1} est donc la matrice complémentaire de $-A$. L'algorithme de Fadeev nous donne le moyen de calculer simultanément les coefficients p_k

---

1. F. R. Gantmacher : *The Theory of Matrices t. I,* pp. 84-89, Chelsea Publishing Company, New York.

J. Lelong-Ferrand, J.-M. Arnaudies : *Cours de mathématiques t. IV,* pp. 57-63, t. I, p. 166, Dunod, Paris, 1977.

et les coefficients matriciels B_k de la matrice complémentaire $B(\lambda)$. Les coefficients p_k sont liés aux traces des puissances A^k par les formules de Newton :

$$\text{tr } A^k = s_k = \sum_{j=1}^{n} \lambda_j^k \qquad k = 1, \ldots, n$$

$$k p_k = s_k - p_1 s_{k-1} - \ldots - p_{k-1} s_1$$

On pose donc successivement :

$$\begin{array}{lll}
A_1 = A & p_1 = \text{tr } A_1 & B_1 = A_1 - p_1 I \\
A_2 = AB_1 & p_2 = \text{tr } \dfrac{A_2}{2} & B_2 = A_2 - p_2 I \\
\vdots & \vdots & \vdots \\
A_k = AB_{k-1} & p_k = \text{tr } \dfrac{A_k}{k} & B_k = A_k - p_k I \\
\vdots & \vdots & \vdots \\
A_n = AB_{n-1} & p_n = \text{tr } \dfrac{A_n}{n} & B_n = A_n - p_n I = 0
\end{array}$$

Effectivement, on retrouve :

$$A_k = A^k - p_1 A^{k-1} - \ldots - p_{k-1} A$$

et :

$$\text{tr } A_k = s_k - p_1 s_{k-1} - \ldots - p_{k-1} s_1 = k p_k$$

ORGANIGRAMME

$$P(\lambda) = \lambda^n - p_1 \lambda^{n-1} - p_2 \lambda^{n-2} - \ldots - p_n$$

B matrice complémentaire de $-A$.

Entrées et initialisation 60, 700

Degré du polynôme n : INPUT N

Coefficients a_{ij} de la matrice A par ligne :

700 DATA $a_{11}, a_{12}, ..., a_{1\,n}$

710 DATA $a_{21}, a_{22}, ..., a_{2\,n}$, etc.

$A(i, j) \longleftarrow a_{ij}$, $B(i, j) \longleftarrow a_{ij}$

$k = 1, ..., n-1$

$TR = 0 : i = 1, ..., n$

$TR = TR + B(i, i)$

$$P(k) = \frac{TR}{k}$$

Algorithme 200-450

$i = 1, ..., n$

$B(i, i) = B(i, i) - P(k)$

$k = n - 1$ — OUI ⟶ résultats

NON

→ $j = 1, ..., n$

$B1(i) = B(i, j)$ ($j^{\text{tième}}$ colonne)

calcul A_{k+1}

$i = 1, ..., n$

$SS = 0 : \ell = 1, ..., n$

$SS = SS + A(i, \ell)\,B1(\ell)$

$B(i, j) = SS$

Résultats 500-660

$i = 1, ..., n$

$j = 1, ..., n$

$B(i, j)$ - matrice complémentaire de $-A$

$k = 1, ..., n$

$P(k)$ - coefficients p_k

PROGRAMME

```
 10 REM *** Polynome caracteristique***
 15 REM *** d'une matrice carree A ***
 20 REM *** Matrice complementaire ***
 25 REM *** B(0) de -A.            ***
 30 REM *** n degre du polynome    ***
 35 REM *** A (n,n)-matrice        ***
 40 PRINT"P(L)=L^n-P1L^(n-1)-...-Pn "
 50 PRINT"B(L)*(L*I-A) = det(L*I - A) "
 60 INPUT "Ordre de la matrice n = ";N
 70 DIM A(N,N),B(N,N),B1(N),P(N)
 80 FOR I=1  TO N
 90 PRINT "  A(";I;",J) = ";
 100 FOR J=1 TO N
 110 READ A(I,J): B(I,J)=A(I,J)
 120 PRINT A(I,J);",";
 130 NEXT
 140 PRINT
 150 NEXT
 190 REM *** Algorithme de FADEEV ***
 200 FOR K = 1 TO N-1
 210 REM *** Trace  ***
 220 TR=0
 230 FOR I=1 TO N
 240 TR = TR + B(I,I)
 250 NEXT I
 260 P(K)=TR/K
 270 REM *** Calcul de la matrice B(K) *
**
 280 FOR I = 1 TO N
 290 B(I,I) = B(I,I) - P(K)
 300 NEXT I
 310 IF K=N-1 THEN 500
 320 REM *** Calcul de la matrice A(K+1)
 ***
 330 FOR J=1 TO N
 340 FOR I=1 TO N
 350 B1(I) = B(I,J)
 360 NEXT I
 370 FOR I=1 TO N
 380 SS=0
```

```
390 FOR L=1 TO N
400 SS=SS + A(I,L)*B1(L)
410 NEXT L
420 B(I,J) = SS
430 NEXT I
440 NEXT J
450 NEXT K
490 REM *** Resultats ***
500 PRINT
510 PRINT"La matrice complementaire B=B
(0) de -A est :"
520 FOR I=1 TO N
530 PRINT "  B(";I;",J) = ";
540 FOR J=1 TO N
550 PRINT B(I,J);",";
560 NEXT
570 PRINT
580 NEXT
590 REM *** Coefficients P(k) ***
600 PRINT:PRINT"Les coefficients P(k),
k=1,...,n sont:"
610 P(N)=0
620 FOR K=1 TO N
630 P(N) = P(N) + A(1,K)*B(K,1)
640 PRINT " P(";K;") = ";P(K)
650 NEXT
660 END
690 REM *** Coefficients de A par ligne
***
700 DATA 1,2,3
710 DATA 2,3,4
720 DATA 3,4,5
```

DÉROULEMENT DU PROGRAMME

Partant de la matrice A, on introduit les coefficients a_{ij} à partir de la ligne 700 :

700 DATA 1,2,3 première ligne

```
710  DATA  2,3,4  deuxième ligne
720  DATA  3,4,5  nième ligne
```

Le programme demande le degré du polynôme $n\,(n=3)$, calcule ensuite les coefficients p_k du polynôme caractéristique :

$$P(\lambda) = \lambda^n - p_1\,\lambda^{n-1} - p_2\,\lambda^{n-2} - \dots - p_n$$

et donne la matrice complémentaire B de $-A$.

On en déduit :

$$A^{-1} = \frac{B}{p_n}, \text{ si } p_n \neq 0$$

$$p_n = (-1)^{n-1} \text{ dét } A$$

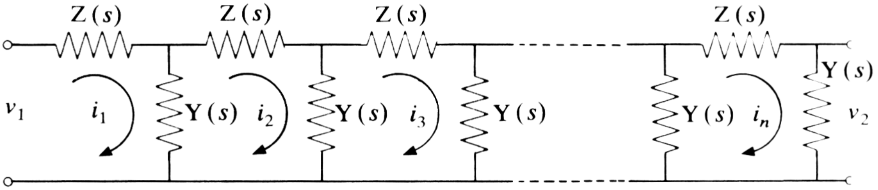
Pour garantir la stabilité de l'algorithme, le degré n ne devrait pas dépasser 10.

Des coefficients a_{ij} entiers sont préférables.

Ensuite on peut se servir du programme Bairstow pour calculer les racines du polynôme.

UNE APPLICATION

Nous allons étudier un circuit électronique comprenant des résistances, des condensateurs ou des inductances, selon le schéma suivant :



Sa fonction de transfert $T = \dfrac{v_2}{v_1}$ présente un intérêt particulier.

Nous notons les n équations du circuit :

$$v_1(s) = (Z + Y)\, i_1(s) - Y\, i_2(s)$$

$$0 = -Y\, i_1(s) + (Z + 2Y)\, i_2(s) - Y\, i_3(s)$$

$$0 = \qquad\qquad\qquad - Y\, i_{n-1}(s) + (Z + 2Y)\, i_n(s)$$

et :

$$T_n = i_n \cdot \frac{Y}{v_1}$$

Nous divisons par Y et trouvons, à l'aide des formules de Cramer :

$$T_n = \frac{Y}{v_1} \frac{\begin{pmatrix} 1+\frac{Z}{Y} & -1 & 0 & . & . & . & \frac{v_1}{Y} \\ -1 & 2+\frac{Z}{Y} & -1 & 0 & . & . & 0 \\ 0 & -1 & 2+\frac{Z}{Y} & -1 & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & -1 & 2+\frac{Z}{Y} & 0 \\ 0 & 0 & 0 & . & 0 & -1 & 0 \end{pmatrix}}{\begin{pmatrix} 1+\frac{Z}{Y} & -1 & 0 & . & . & . & . & 0 \\ -1 & 2+\frac{Z}{Y} & -1 & 0 & . & . & . & 0 \\ 0 & -1 & 2+\frac{Z}{Y} & -1 & 0 & . & . & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ 0 & 0 & . & . & . & -1 & 2+\frac{Z}{Y} & -1 \\ 0 & 0 & . & . & . & 0 & -1 & 2+\frac{Z}{Y} \end{pmatrix}}$$

$$
= \frac{\frac{Y}{v_1}(-1)^{n-1}\frac{v_1}{Y}(-1)^{n-1}}{\text{dét}\left(I\frac{Z}{Y}-A\right)}
$$

$$
A = \begin{pmatrix} -1 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{pmatrix}
$$

Il faut donc calculer le polynôme caractéristique de la matrice A. Toutes les valeurs propres sont d'ailleurs négatives et les zéros du polynôme également (A est définie négative). Quoiqu'il existe un algorithme très simple pour calculer le polynôme caractéristique d'une matrice tridiagonale, nous allons appliquer la méthode de Fadeev.

Remarque

Une matrice tridiagonale possède des éléments non nuls uniquement dans la diagonale principale et dans les deux diagonales adjacentes.

Discutons encore le cas $n=3$. On trouve quand $Z=R$, $Y=Cs$:

$$
T_3 = \frac{1}{R^3C^3s^3 + 5R^2C^2s^2 + 6RCs + 1}
$$

et les zéros du dénominateur sont tous situés sur l'axe négatif réel. Admettons une tension sinusoïdale à l'entrée. On peut donc remplacer s par $j\omega$ et arriver à :

$$
\frac{1}{T_3} = 1 - 5R^2C^2\omega^2 + j(6RC\omega - R^3C^3\omega^3)
$$

Si la partie imaginaire $6RC\omega - R^3C^3\omega^3$ disparaît, on a un déphasage de $-\pi$ entre v_1 et v_2. La pulsation correspondante ω_0 est alors :

$$
\omega_0 = \sqrt{\frac{6}{RC}}
$$

et le rapport $\frac{v_1}{v_2}$ devient :

$$\frac{v_1}{v_2} = 1 - 30 = -29$$

Pour réaliser un oscillateur, il faut donc ajouter un amplificateur qui assure un gain égal à $\frac{29}{1}$ pour cette fréquence $\frac{\omega_0}{2\pi}$.

La fréquence de coupure à -3 dB $f_c = \frac{\omega_c}{2\pi}$ suffit à la relation :

$$|T_3(j\omega_c)|^2 = \frac{1}{(1 + 26\,R^2C^2\,\omega_c^2 + 13\,R^4C^4\,\omega_c^4 + R^6C^6\,\omega_c^6)} = \frac{1}{2}$$

ou :

$$x^6 + 13\,x^4 + 26\,x^2 - 1 = 0 \qquad x = RC\,\omega_c$$

Notre programme Bairstow va nous livrer tous les zéros, et un calcul rapide nous montre déjà :

$$\omega_c \sim \frac{1}{\sqrt{26}\,RC}$$

EXEMPLES NUMÉRIQUES

EXEMPLE 1

```
P(L)=L^n-P1L^(n-1)-...-Pn
B(L)*(L*I-A) = det(L*I - A)
Ordre de la matrice n = 3
   A(1 ,J) = 1        2        3
   A(2 ,J) = 2        3        4
   A(3 ,J) = 3        4        5
```

```
La matrice complementaire B=B(0) de -A est :
  B(1 ,J) = -1      2       -1
  B(2 ,J) = 2       -4      2
  B(3 ,J) = -1      2       -1

Les coefficients P(k), k=1,...,n sont:
 P(1 ) = 9
 P(2 ) = 6
 P(3 ) = 0
```

EXEMPLE 2

```
P(L)=L^n-P1L^(n-1)-...-Pn
B(L)*(L*I-A) = det(L*I - A)
Ordre de la matrice n = ? 5

A(1,J) =   1   1/2  1/3  1/4  1/5
A(2,J) = 1/2 1/3  1/4  1/5  1/6
A(3,J) = 1/3 1/4  1/5  1/6  1/7
A(4,J) = 1/4 1/5  1/6  1/7  1/8
A(5,J) = 1/5 1/6  1/7  1/8  1/9
```

Pour cette matrice de Hilbert H_5 on trouve un résultat complètement aberrant :

```
Les coefficients P(k), k=1,...,n sont:
 P(1 ) = 1.78730159
 P(2 ) = -.347591176
 P(3 ) = 3.83508337E-03
 P(4 ) = -1.15305054E-06
 P(5 ) = -2.1658249E-10
```

valeur exacte : $3{,}749 \cdot 10^{-12}$

Considérons donc $2\,520\,H_5$:

EXEMPLE 3

```
P(L)=L^n-P1L^(n-1)-...-Pn
B(L)*(L*I-A) = det(L*I - A)
Ordre de la matrice n = ? 5
  A( 1 ,J) =  2520 , 1260 , 840 , 630 ,
 504 ,
  A( 2 ,J) =  1260 , 840 , 630 , 504 ,
 420 ,
  A( 3 ,J) =  840 , 630 , 504 , 420 ,
 360 ,
  A( 4 ,J) =  630 , 504 , 420 , 360 ,
 315 ,
  A( 5 ,J) =  504 , 420 , 360 , 315 ,
 280 ,


La matrice complementaire B=B(0) de -A e
st :
  B( 1 ,J) =  3780 ,-45360 , 158760 ,
-211680 , 95256 ,
  B( 2 ,J) = -45360 , 725760 ,-2857680 ,
 4064256 ,-1905120 ,
  B( 3 ,J) =  158760 ,-2857680 ,
 12002256 ,-17781120 , 8573040 ,
  B( 4 ,J) = -211680 , 4064256 ,
-17781120 , 27095040 ,-13335840 ,
  B( 5 ,J) =  95256 ,-1905120 , 8573040
,-13335840 , 6667920 ,

Les coefficients P(k), k=1,...,n sont:
 P( 1 ) =  4504
 P( 2 ) = -2207343
 P( 3 ) =  61372872
 P( 4 ) = -46494756
 P( 5 ) =  381024
```

et le résultat est exact.

EXEMPLE 4

```
P(L)=L^n-P1L^(n-1)-...-Pn
B(L)*(L*I-A) = det(L*I - A)
Ordre de la matrice n = 6
  A(1 ,J) = -1       1        0        0        0        0
  A(2 ,J) = 1        -2       1        0        0        0
  A(3 ,J) = 0        1        -2       1        0        0
  A(4 ,J) = 0        0        1        -2       1        0
  A(5 ,J) = 0        0        0        1        -2       1
  A(6 ,J) = 0        0        0        0        1        -2

La matrice complementaire B=B(0) de -A est :
  B(1 ,J) = 6        5        4        3        2        1
  B(2 ,J) = 5        5        4        3        2        1
  B(3 ,J) = 4        4        4        3        2        1
  B(4 ,J) = 3        3        3        3        2        1
  B(5 ,J) = 2        2        2        2        2        1
  B(6 ,J) = 1        1        1        1        1        1

Les coefficients P(k), k=1,...,n sont:
 P(1 ) = -11
 P(2 ) = -45
 P(3 ) = -84
 P(4 ) = -70
 P(5 ) = -21
 P(6 ) = -1
```

EXEMPLE 5

```
P(L)=L^n-P1L^(n-1)-...-Pn
B(L)*(L*I-A) = det(L*I - A)
Ordre de la matrice n = 3
  A(1 ,J) = -1       1        0
  A(2 ,J) = 1        -2       1
  A(3 ,J) = 0        1        -2

La matrice complementaire B=B(0) de -A est :
  B(1 ,J) = 3        2        1
  B(2 ,J) = 2        2        1
  B(3 ,J) = 1        1        1

Les coefficients P(k), k=1,...,n sont:
 P(1 ) = -5
 P(2 ) = -6
 P(3 ) = -1
```

L'exemple 4 nous livre :

$$T_6 = \frac{1}{R^6 C^6 s^6 + 11 R^5 C^5 s^5 + 45 R^4 C^4 s^4 + 84 R^3 C^3 s^3 + 70 R^2 C^2 s^2 + 21 RCs + 1}$$

fonction de transfert du déphaseur 6 RC; l'exemple 5 donne T_3.

10. STABILITÉ D'UN POLYNÔME

INITIATION AU PROBLÈME

Considérons une boucle à verrouillage de phase d'ordre 3. Elle se compose de quatre parties (voir le Schéma 10.1) :

- Un comparateur de phase CP chargé d'élaborer la différence entre les phases instantanées $\varphi = \varphi_i - \varphi_r$ de deux tensions v_i et v_r.
- Un filtre passe bas FP chargé de filtrer la composante haute fréquence (par exemple deux filtres actifs montés en cascade).
- Un oscillateur commandé en tension (VCO) délivrant une tension de sortie v_r, de pulsation ω_r proportionnelle à sa tension d'entrée v_2, sa pulsation centrale étant ω_0.
- Un intégrateur immatériel : $\varphi_r = \int \omega_r \, dt$.

La fonction de transfert F de notre filtre passe bas prend la forme :

$$F(p) = \left(\frac{1 + \tau_2 \, p}{\tau_1 \, p} \right)^2$$

Supposons la boucle au repos verrouillée sur ω_0 et appliquons à l'instant $t = 0$ un échelon de fréquence :

$$\omega_i = \omega_0 + \Delta\Omega$$

En régime de poursuite ($\Delta\Omega$ de faible valeur), le système est régi par l'équation différentielle linéarisée :

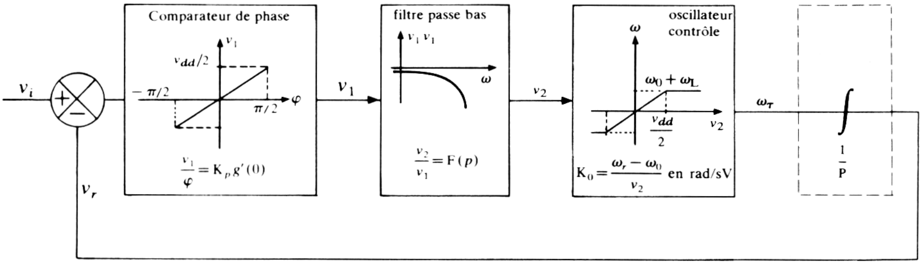


Schéma 10.1.

$$\frac{d^3\varphi}{dt^3} + F^2 K \left(\frac{d^2\varphi}{dt^2} + \frac{2}{\tau_2}\frac{d\varphi}{dt} + \frac{1}{\tau_2^2}\varphi\right) = 0$$

$$F = \frac{\tau_2}{\tau_1}, \; K = K_p K_0 g'(0)$$

transmittance statique de la boucle.

La boucle ne peut reverrouiller que si toutes les solutions φ de l'équation tendent vers zéro quand t tend vers l'infini. Pour cela, il faut et il suffit que la partie réelle des racines de son polynôme caractéristique :

$$p(\lambda) = \lambda^3 + F^2 K \left(\lambda + \frac{1}{\tau_2}\right)^2$$

soit négative.

Passons au cas général et regardons le polynôme :

$$P(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n, \; a_0 \neq 0$$

Nous voulons donc savoir si tous les zéros appartiennent au demi-plan gauche (voir la Figure 10.2).

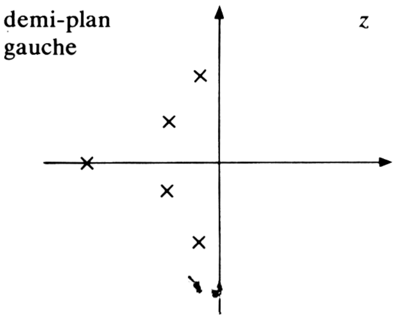


Figure 10.2 : Demi-plan gauche.

MÉTHODE

Nous appliquons l'algorithme de Routh [1] et nous posons :

$$b_k^{(0)} = a_{2k}, \; k = 0, 1, \ldots, n1, \; n1 = \text{int}\left(\frac{n}{2}\right)$$

$$b_k^{(1)} = a_{2k+1}, \; k = 0, 1, \ldots, \text{int}\left(\frac{n-1}{2}\right)$$

et :

$$b_{n1}^{(1)} = a_{n+1} = 0 \qquad \text{si } n \text{ est pair}$$

Pour $i = 1, \ldots, n$, on calcule :

$$d_i := \frac{b_0^{(i)}}{b_0^{(i-1)}}, \; b_k^{(i+1)} := d_i \, b_{k+1}^{(i-1)} - b_{k+1}^{(i)},$$

$$k = 0, \ldots, n1 - 1,$$

1. P. Henrici : *Applied and Computational Complex Analysis*, Vol. I, II. Wiley, New York, 1974-1977.

et on pose :

$$b_{n1}^{(i+1)}=0$$

Le polynôme est stable si *(et seulement si)* toutes les fractions d_i existent et sont positives.

Notre exemple nous donne :

$n=3$ d_i	$k=0$	$k=1=n1$	
	$\frac{F^2 K}{\tau_2^2}$	$F^2 K$	$i=0$
$2\tau_2$	$\frac{2F^2 K}{\tau_2}$	1	$i=1$
$\tau_2^2-\frac{\tau_2}{2F^2 K}$	$2\tau_2 F^2 K-1$	0	$i=2$
$\frac{\tau_2}{(2F^2 K)}$	$\tau_2^2-\frac{\tau_2}{2F^2 K}$	0	$i=3$

Le polynôme est donc stable si *(et seulement si) :*

$$d_2>0 \text{ ou } 2F^2 K\tau_2>1$$

ORGANIGRAMME

$$\boxed{P(x)=a_0+a_1 x+a_2 x^2+\ldots+a_n x^n}$$

Entrées	Degré du polynôme : $N \longleftarrow n$
40-80	(coefficients $a_0, a_1, a_2, \ldots, a_n$
250	250 DATA ., ., ..., .)
	$k=0, 1, \ldots, N : A(k) \longleftarrow a_k$, n pair : $A(N+1)=0$

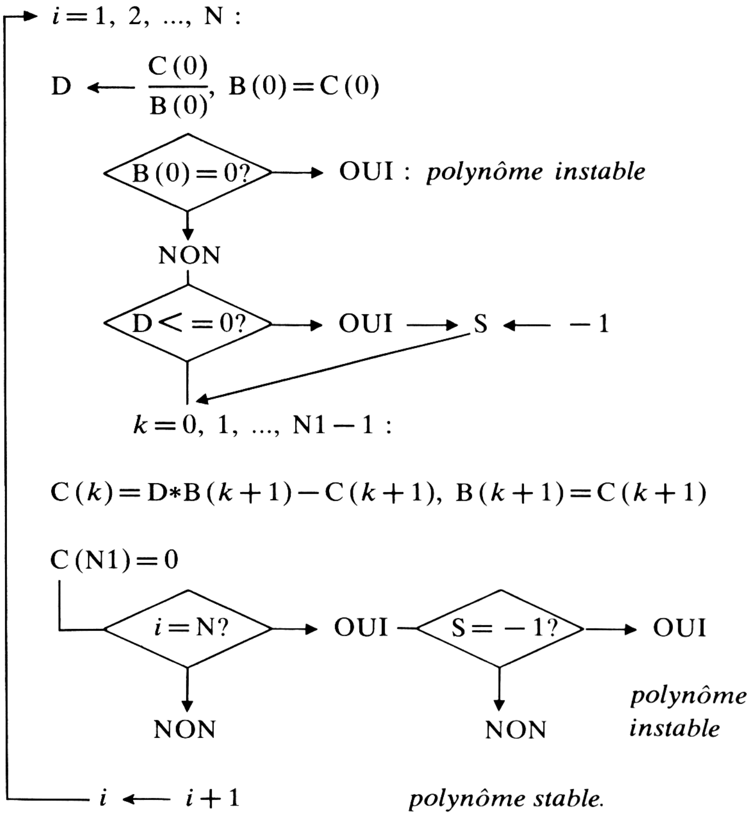
Initialisation $N1 \longleftarrow \text{int}\left(\frac{N}{2}\right), S \longleftarrow 0$

100-135 $k = 0, 1, \ldots, N1$:

$B(k) \longleftarrow A(2k), C(k) \longleftarrow A(2k+1)$

Algorithme

140-200



Variables

N Degré du polynôme.

N1 $\text{int}\left(\frac{N}{2}\right)$.

D Fraction décisive de l'algorithme.

S Variable de la stabilité.

I, K Variables de boucles locales.

PROGRAMME

```
 10 REM *** Algorithme de ROUTH - Stabil
ite d'un Polynome ***
 20 REM *** Degre du Polynome - coeffici
ents ***
 30 INPUT "Degre n = ";N
 40 DIM A(N+1)
 50 FOR K=0 TO N
 60 READ A(K)
 65 NEXT
 70 IF N = 2*INT(N/2) THEN A(N+1) = 0
 80 IF A(0) = 0 THEN 300
 90 REM *** Algorithme ***
 100 N1 = INT(N/2) : S=0
 110 DIM B(N1),C(N1)
 120 FOR K=0 TO N1
 130 B(K) = A(2*K): C(K) = A(2*K+1)
 135 NEXT
 140 FOR I = 1 TO N
 150 D = C(0)/B(0): PRINT D: B(0)=C(0):
IF B(0)=0 THEN 300
 155 IF D<=0 THEN S=-1
 160 FOR K=0 TO N1-1
 170 C(K) = D*B(K+1) - C(K+1): B(K+1)=C(
K+1)
 180 NEXT K
 190 C(N1)=0
 200 NEXT I
 210 IF S=-1 THEN 300
 220 PRINT "Polynome stable, partie reel
le des racines < 0."
 240 REM *** n+1 coefficients du Polynom
e a(0)+a(1)x+...+a(n)x^n ***
 250 DATA 1,2,0,3,5,6
 260 END
```

```
 300 PRINT "Polynome instable, Partie re
elle d'une ou Plusieurs racines >=0 "
 310 END
```

DÉROULEMENT DU PROGRAMME

Partant du polynôme :

$$P(x) = a_0 + a_1 x + \dots + a_n x^n$$

on introduit les coefficients a_0, a_1, a_2, ..., a_n en ligne 250, n étant le degré du polynôme.

250 DATA ., ., ., .

$n+1$ nombres

Le programme demande le degré du polynôme :

30 INPUT "Degré $n=$" ; N

Il calcule ensuite les fractions décisives d_i de l'algorithme et les affiche sur l'écran. Si les n valeurs affichées sont strictement positives, le polynôme est stable, dans le cas contraire il est instable.

EXEMPLES NUMÉRIQUES

Exemple 1

```
P(x) = 1+2x+3x^3+5x^4+6x^5
```

```
 250 DATA 1,2,0,3,5,6
Degre n = ? 5
 2
-1.5
 2.83333333
-2.39215686
 1.05882353
Polynome instable, Partie reelle d'une o
u Plusieurs racines >=0
```

Exemple 2

```
P(x) = 1+4x+6x^2+4x^3+x^4
```

```
 250 DATA 1,4,6,4,1
Degre n = ? 4
 4
 5
 .8
 .2
Polynome stable, Partie reelle des racin
es < 0.
```

Exemple 3

```
P(x) = -12+4x+15x^2-5x^3-3x^4+x^5
```

```
 250 DATA -12,4,15,-5,-3,1
Degre n = ? 5
-.333333333
 0
Polynome instable, Partie reelle d'une o
u Plusieurs racines >=0
```

11. ZÉROS D'UN POLYNÔME

INITIATION AU PROBLÈME

Comme on a pu le voir précédemment, en cherchant les valeurs propres d'une matrice A ou la fonction de transfert d'un déphaseur nRC, on était ramené à résoudre l'équation algébrique :

$$\text{dét}(xI - A) = p(x) = x^n + a_{n-1} x^{n-1} + \ldots + a_0 = 0$$

La résolution d'une équation différentielle linéaire d'ordre n à coefficients constants :

$$y^{(n)} + a_{n-1} y^{(n-1)} + \ldots + a_1 y' + a_0 y = 0$$

nous conduit d'ailleurs au même polynôme caractéristique.

La méthode de Bairstow permettra le calcul explicite des racines d'un polynôme quelconque. Si le degré de $p(x)$ devient trop grand, une certaine instabilité est à craindre. En ce cas-là, on attaque le problème des valeurs propres d'une matrice directement en se servant des méthodes directes de Jacobi, Householder [1].

MÉTHODE

Nous considérons donc une équation non linéaire :

$$f(x) = 0$$

1. J. Legras : *Méthodes et Techniques de l'analyse numérique*, Dunod, Paris, 1971.

où f est un polynôme p de degré n :

$$p(x)=a_n x^n + a_{n-1} x^{n-1} + \ldots + a_0, \ a_n \neq 0$$

Nous supposons les coefficients a_k réels.

p possède n racines réelles ou complexes et les zéros non réels sont conjugués deux à deux :

$$p(z_0)=0, \ z_0 = x_0 + jy_0 \Longleftrightarrow p(\overline{z_0})=0, \ \overline{z_0} = x_0 - jy_0$$

Pour que z_0 soit une racine de multiplicité m, il faut et il suffit que :

$$p^{(m-1)}(z_0)=0 \text{ et } p^{(m)}(z_0)\neq 0, \ \left(p^{(m)}=\frac{d^m p}{dx^m}\right)$$

On peut donc décomposer p en un produit :

$$p(x)=a_n (x^2+p_1 x+q_1)^{m_1} \ldots (x^2+p_r x+q_r)^{m_r} \cdot (x-x_{r+1})^{m_{r+1}} \ldots \cdot (x-x_s)^{m_s}$$

$$p_k, \ q_k, \ x_k \text{ réels, } 2(m_1+\ldots+m_r)+m_{r+1}+\ldots+m_s=n$$

Divisons le polynôme par le trinôme x^2+px+q :

$$p(x)=(x^2+px+q)\,p_1(x)+\mathrm{R}(p, q)\,x+\mathrm{S}(p, q)$$

$$p_1(x)=b_n x^{n-2}+b_{n-1} x^{n-3}+\ldots+b_2$$

Il faut choisir p et q de façon que $\mathrm{R}(p, q)$ et $\mathrm{S}(p, q)$ s'annulent (paramètre PS précision de solution).

Pour trouver un couple (p, q) convenable, on applique la méthode de Newton. En partant des valeurs initiales p_0, q_0 (paramètres P, Q), on obtient deux nouvelles approches p_1, q_1 et ensuite par itération p_n, q_n (paramètres NI : nombre d'itérations, PN précision) :

$$\begin{pmatrix} p_n \\ q_n \end{pmatrix}=\begin{pmatrix} p_{n-1} \\ q_{n-1} \end{pmatrix}-\frac{1}{\mathrm{R}_p \mathrm{S}_q - \mathrm{S}_p \mathrm{R}_q}\begin{pmatrix} \mathrm{S}_q & -\mathrm{R}_q \\ -\mathrm{S}_p & \mathrm{R}_p \end{pmatrix}\begin{pmatrix} \mathrm{R} \\ \mathrm{S} \end{pmatrix}_{|(p_{n-1}, q_{n-1})}$$

$$\mathrm{S}_q=\frac{\partial \mathrm{S}}{\partial q}(p_{n-1}, q_{n-1}), \text{ etc.}$$

Nous adaptons le schéma de Horner et nous trouvons :

$$b_n = a_n$$

$$b_{n-1} = a_{n-1} - pb_n$$

$$b_k = a_k - pb_{k+1} - qb_{k+2}, \ k = n-2, \ldots, 0$$

Reste à calculer les dérivées S_p, S_q, R_p, R_q.
En évaluant des formules de récurrence analogues :

$$c_n = b_n$$

$$c_{n-1} = b_{n-1} - pc_n$$

$$c_k = b_k - pc_{k+1} - qc_{k+2}, \ k = n-2, \ldots, 2$$

$$c_1 = -pc_2 - qc_3$$

on vérifie :

$$R(p, q) = b_1, \ S(p, q) = b_0 + pb_1$$

$$R_p(p, q) = -c_2, \ R_q(p, q) = -c_3$$

$$S_p(p, q) = -c_1 - pc_2, \ S_q(p, q) = -c_2 - pc_3$$

et :

$$\Delta p = \frac{-S_q R + R_q S}{R_p S_q - S_p R_q} = \frac{c_2 b_1 - c_3 b_0}{c_2^2 - c_1 c_3}$$

$$\Delta q = \frac{S_p R - R_p S}{R_p S_q - S_p R_q} = \frac{c_2 b_0 - c_1 b_1}{c_2^2 - c_1 c_3}$$

Après quelques itérations, on est ramené à une solution approchée $\bar{p}$, $\bar{q}$ des équations :

$$R(p, q) = 0, \quad S(p, q) = 0$$

D'où :

$$p(x) \approx p_1(x)(x^2 + \bar{p}x + \bar{q})$$

et :

$$z_{1,2} = -\frac{\bar{p}}{2} \pm \sqrt{\frac{\bar{p}^2}{4} - \bar{q}}$$

sont deux racines du polynôme p.

Le même procédé se déroule une autre fois. On pose :

$$a_{k-2} = b_k, \quad k = 2, \ldots, n$$
$$n \longleftarrow n - 2$$

et on divise (si $n > 2$) p_1 par le trinôme :

$$x^2 + px + q \ldots$$

Finalement, on arrive à un polynôme restant de degré 2 ou 1 dont les zéros sont calculés directement.

Comme toutes les racines sont obtenues d'une manière approchée, la précision se détériore au cours de l'algorithme (les termes R et S ne sont pas exactement nuls). Les dernières racines seront donc moins précises que les premières.

Une amélioration des résultats est pourtant facile. On reprend la méthode de Bairstow au départ en choisissant comme valeurs initiales les coefficients $\bar{p}$, $\bar{q}$, du dernier trinôme obtenus à la première phase.

ORGANISATION DU PROGRAMME

Initialisation

1700 REM *** coefficients $a(k)$ du polynôme

$$a(n)x^n + \ldots + a(0)$$

1710 DATA A(N), A(N−1), ..., A(0)

Définition des constantes

1030 — $PS = 1E-4$: précision de solution du système non linéaire $R(P, Q) = 0$, $S(P, Q) = 0$

$PN = 1E-9$: précision de la méthode de Newton

$NI = 70$: nombre d'itérations

Introduction des variables

1050 — Degré du polynôme : N

1130-1140 — Valeurs initiales P, Q du trinôme

$X^2 + PX + Q$

Remarque

Si les valeurs PS, PN et NI ne conviennent pas, l'utilisateur peut introduire ses propres valeurs. En cas de racines multiples, le choix de ces valeurs, et surtout des valeurs initiales, demande de temps en temps un peu de doigté.

Résultats 1150-1370 — Affichage des racines du polynôme

Boucle du calcul 1380-1420

Les algorithmes — 1500-1580 schéma de Horner

1590-1690 méthode de Newton-Itération

Résolution du système non linéaire

$R(P, Q) = S(P, Q) = 0$

PROGRAMME

```
 1000 REM *** Methode de BAIRSTOW ***
 1005 REM *** Racines d'un Polynome ***
 1010 REM *** a coefficients reels ***
 1015 REM *** PS: Precision de solution
du systeme R(p,q) =0, S(p,q) =0 ***
 1020 REM *** PN: Precision de la method
e de NEWTON (linearisation) ***
 1030 PS = 1E-4: PN = 1E-9 : NI=70:NJ=NI

 1035 REM *** n degre du Polynome P(x)=a
(n)x^n+a(n-1)x^(n-1)+...+a(0) ***
 1040 REM *** P(x)=(x^2+Px+q)*P1(x) +  R
(p,q)x + S(p,q) ***
 1050 INPUT "Degre du Polynome n = ";N
 1060 REM *** Introduction des
           coefficients ***
 1070 DIM A(N),B(N),C(N)
 1080 PRINT "Coefficients a(n),a(n-1),..
.a(0) : "
 1090 FOR K = N TO 0 STEP -1
 1100 READ A(K): PRINT " ";A(K),
 1110 NEXT
 1120 REM *** Introduction des valeurs
 initiales p,q du trinome x^2+Px+q***
 1125 PRINT
 1130 INPUT " p = ";P
 1140 INPUT " q = ";Q
 1150 PRINT:PRINT "Racines du Polynome"
 1160 PRINT "Methode de BAIRSTOW "
 1170 IF N > 2 THEN 1500
 1200 IF N = 1 THEN PRINT -A(0)/A(1) : E
ND
 1210 P1 ='A(1)/A(2) : Q1 = A(0)/A(2)
 1300 RE = -P1/2: IM = RE*RE - Q1: IF IM
<0 THEN 1350
```

```
1310 PRINT RE + SQR (IM)
1320 PRINT RE - SQR (IM)
1330 GOTO 1380
1350 IM = SQR(-IM)
1360 PRINT RE;" +j*";IM
1370 PRINT RE;" -j*";IM
1380 N=N-2: IF N=0 THEN END
1390 FOR K = 0 TO N
1400 A(K) = B(K+2)
1410 NEXT
1420 GOTO 1170
1490 REM *** Methode de BAIRSTOW ***
1500 NI=0
1510 B(N) = A(N): C(N)= A(N)
1520 B(N-1) = A(N-1) - P*B(N)
1530 C(N-1) = B(N-1) - P*C(N)
1540 FOR K = N-2 TO 0 STEP -1
1550 B(K) = A(K) -P*B(K+1) -Q*B(K+2)
1560 C(K) = B(K) -P*C(K+1) -Q*C(K+2)
1570 NEXT
1580 C(1) = C(1) - B(1)
1590 REM *** Iteration  ***
1600 D = C(2)*C(2) - C(1)*C(3)
1610 P1 = P +(C(2)*B(1) - C(3)*B(0))/D
1620 Q1 = Q +(C(2)*B(0) - C(1)*B(1))/D
1630 IF (ABS(B(1)) < PS ) AND (ABS(B(0)
+P*B(1)) < PS) THEN 1300
1640 IF (ABS(P1-P)<PN) AND (ABS(Q1-Q)<P
N) THEN 1670
1650 NI=NI+1: IF NI>NJ THEN 1690
1660 P=P1:Q=Q1: GOTO 1520
1670 PRINT"Precision ";PS;" non obtenue
, choisissez d'autres valeurs initiales"
1680 END
1690 PRINT"Pas de convergence apres ";N
J;" iterations, augmentez NI.":END
1700 REM *** a(n), a(n-1),...., a(0) ***
1710 DATA 1,0,0,0,0,0,0,0,-1
```

EXEMPLES NUMÉRIQUES

Racines complexes

```
Degre du Polynome n = ? 8
Coefficients a(n),a(n-1),...a(0) :
  1        0        0        0        0
  0        0        0       -1
 p = ? 1
 q = ? 1

Racines du Polynome
Methode de BAIRSTOW
 1
-1
-1.20914692E-07  +j* 1.00000012
-1.20914692E-07  -j* 1.00000012
 .707106875  +j* .707106988
 .707106875  -j* .707106988
-.707107753  +j* .707108231
-.707107753  -j* .707108231
```

Racines multiples

```
Degre du Polynome n = ? 5
Coefficients a(n),a(n-1),...a(0) :
  324     -351      108      6       -8
  1
 p = ? 1
 q = ? 1

Racines du Polynome
Methode de BAIRSTOW
 .338163031
 .327236688
 .334597401  +j* 7.77022464E-03
 .334597401  -j* 7.77022464E-03
-.249999126
```

```
Degre du Polynome n = ? 5
Coefficients a(n),a(n-1),...a(0) :
  324    -351      108      6       -8
  1
 p = ? -.66
 q = ? .111

Racines du Polynome
Methode de BAIRSTOW
 .333227846  +J* 7.94540729E-03
 .333227846  -J* 7.94540729E-03
 .344776092
 .322312755
-.250000313

 Valeurs exactes: 1/3, 1/3, 1/3, 1/3, -1/4


 1030 PS = 1E-7:PN = 1E-11: NI=99: NJ=NI
Degre du Polynome n = ? 5
Coefficients a(n),a(n-1),...a(0) :
  324    -351      108      6       -8
  1
 p = ? -.66
 q = ? .111

Racines du Polynome
Methode de BAIRSTOW
 .333326719  +J* 2.01676662E-03
 .333326719  -J* 2.01676662E-03
 .336167462
 .330525753
-.250000001
```

Le lecteur est invité à comparer le résultat avec celui obtenu par le programme « Résolution d'une équation non linéaire $f(x)=0$ ».

Nous reprenons l'étude des fonctions de transfert (voir programme Fadeev).

Déphaseur 6 RC

```
Degre du Polynome n = ? 6
Coefficients a(n),a(n-1),...a(0) :
  1        11        45        84        70
  21       1
 p = ? 1
 q = ? 1

Racines du Polynome
Methode de BAIRSTOW
-.502978503
-1.29079023
-.0581176228
-2.24108784
-3.13613975
-3.77090322
```

Le régime transitoire risque d'être long à cause de la racine $-0{,}058$ très proche de zéro.

Fréquence de coupure à -3 dB

```
Degre du Polynome n = ? 6
Coefficients a(n),a(n-1),...a(0) :
  1        0         13        0         26
  0        -1
 p = ? 1
 q = ? 1

Racines du Polynome
Methode de BAIRSTOW
 .194286
-.194286
-5.2695534E-10  +j* 1.58695514
-5.2695534E-10  -j* 1.58695514
 1.27591236E-10  +j* 3.24335019
 1.27591236E-10  -j* 3.24335019
```

On trouve donc :

$$\omega_c = \frac{0,1943}{RC}$$

ANNEXE

```
 10 REM ZEROS D'UN POLYNOME A COEFFICIEN
TS COMPLEXES
 20 REM A(N)X^N+A(N-1)X^(N-1)+....A(0),
A(0)<>0
 30 REM COEFFICIENTS- REA(K), IMA(K) VOI
R DATA
 40 INPUT"DEGRE DU POLYNOME N= ";N
 45 REM N>=2
 50 INPUT"PRECISION P= ";P
 60 INPUT"NOMBRE D'ITERATIONS M1: ";M1
 70 REM DATA REA(0), IMA(0).......RE A(N
-1),IMA(N-1),REA(N),IA(N)
 80 DATA 1,1,-1,0,-1,-1,1,0
 100 DIM RA(N),IA(N),RM(N,N),IM(N,N),RX(
N),IX(N)
 110 FOR K=0 TO N
 120 READ RA(K),IA(K)
 130 NEXT
 140 RO=(RA(0)*RA(0)+IA(0)*IA(0))^(N/2)
 200 FOR K=1 TO N
 210 RX(K)=RO*COS(2*PI*K/N):IX(K)=RO*SIN
(2*PI*K/N)
 230 NEXT
 500 REM ITERATIONS
 510 M=0
 520 S=0
 530 GOSUB 1000
 540 FOR I=1 TO N
 550 RX=RX(I):IX=IX(I)
 560 GOSUB 2000
 570 GOSUB 3000
 580 S=S+ABS(RX(I)-RY)+ABS(IX(I)-IY)
 585 RX(I)=RY:IX(I)=IY
```

```
 590 NEXT
 600 IF S<P THEN 800:REM PRECISION OBTEN
UE
 640 M=M+1: IF M=M1 THEN 700
 650 GOTO 520
 700 PRINT"PRECISION ";P;" NON OBTENUE"
 710 PRINT"ERREUR:";S
 720 PRINT"DERNIERE APPROCHEE:"
 730 GOTO 820
 800 REM RESULTATS
 810 PRINT"SOLUTION:"
 820 FOR K=1 TO N
 830 PRINT "ZERO No ";K;"=: "; INT(RX(K)
/P+.5)*P;" + i*";INT(IX(K)/P+.5)*P
 840 NEXT
 850 END
 1000 REM MATRICE DES DIFFERENCES
 1100 FOR J=2 TO N
 1110 FOR K=1 TO J-1
 1120 RM(J,K)=RX(J)-RX(K)
 1130 IM(J,K)=IX(J)-IX(K)
 1150 RM(K,J)=-RM(J,K)
 1160 IM(K,J)=-IM(J,K)
 1170 NEXT
 1175 RM(J,J)=RA(N):IM(J,J)=IA(N)
 1180 NEXT
 1185 RM(1,1)=RA(N):IM(1,1)=IA(N)
 1190 RETURN
 2000 REM VALEUR DU POLYNOME
 2100 RZ=RA(N):IZ=IA(N)
 2110 FOR K=N TO 1 STEP -1
 2120 RU=RX*RZ-IX*IZ+RA(K-1)
 2130 IZ=RX*IZ+IX*RZ+IA(K-1)
 2140 RZ=RU
 2150 NEXT
 2160 RETURN
 3000 REM  PRODUIT DES APPROCHEES
 3100 RM=RM(I,1)
 3110 IM=IM(I,1)
 3120 FOR K=1 TO N-1
 3130 RN=RM*RM(I,K+1)-IM*IM(I,K+1)
```

```
 3140 IM=RM*IM(I,K+1)+IM*RM(I,K+1)
 3150 RM=RN
 3160 NEXT
 3165 IF IM=0 THEN RY=RX-RZ/RM:IY=IX-IZ/
RM:RETURN
 3170 IF RM=0 THEN RY=RX-IZ/IM:IY=IX+RZ/
IM:RETURN
 3175 Q1=IM/RM:Q2=1/(RM+IM*Q1):Q3=1/(IM+
RM/Q1)
 3180 RY=RX-RZ*Q2-IZ*Q3
 3190 IY=IX-IZ*Q2+RZ*Q3
 3200 RETURN
```

12. RÉSOLUTION D'UN SYSTÈME D'ÉQUATIONS LINÉAIRES

INITIATION AU PROBLÈME

Commençons par un exemple, le *Pont de Wheastone déséquilibré* (voir la Figure 12.1). Le problème est classique en électricité. Il s'agit de déterminer les courants dans toutes les branches de ce circuit. La méthode des mailles indépendantes permet d'écrire :

$$\begin{cases} R_1 i_1 + g(i_1 - i_2) + R_3(i_1 - i_3) = 0 \\ g(i_2 - i_3) + R_2 i_2 + R_4(i_2 - i_3) = 0 \\ \tau i_3 + R_3(i_3 - i_1) + R_4(i_3 - i_2) = E \end{cases}$$

D'où :

$$\begin{cases} i_1[R_1 + g + R_3] + i_2[-g] + i_3[-R_3] = 0 \\ i_1[0] + i_2[g + R_2 + R_4] + i_3[-g - R_4] = 0 \\ i_1[-R_3] + i_2[-R_4] + i_3[\tau + R_3 + R_4] = E \end{cases}$$

On a donc trois équations à trois inconnues.

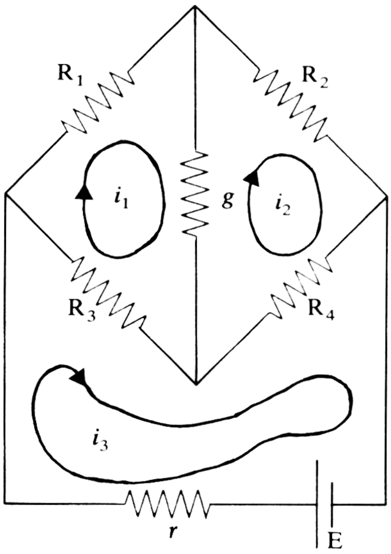


Figure 12.1.

MÉTHODE

Nous allons résoudre le système linéaire :

$$-b+Ax=0 \qquad A\ (n,\ n)-\text{matrice, dét } A\neq 0$$

par un procédé d'orthogonalisation.

Pour que x soit la solution de $Ax=b$, il faut et il suffit que le vecteur $(-1,\ x_1,\ \ldots,\ x_n)$ soit orthogonal à $(b_i,\ a_{i1},\ a_{i2},\ \ldots,\ a_{in})$, $i=1,\ \ldots,\ n$.

On applique donc le procédé bien connu de Schmidt à la matrice :

$$\begin{pmatrix} 0 & b_1 & b_2 & \ldots & b_n \\ b_1 & a_{11} & a_{12} & \ldots & a_{1n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ b_n & a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix} =: \begin{pmatrix} A(0) \\ A(1) \\ \vdots \\ A(N) \end{pmatrix}$$

en commençant par la dernière ligne A (N) :

$$(b_n\, a_{n1}\, a_{n2} \dots a_{nn}) \longleftarrow \frac{1}{|\mathrm{A}(\mathrm{N})|}\,\mathrm{A}(\mathrm{N})$$

A (N − 1) est d'abord remplacé par :

$$\mathrm{A}(\mathrm{N}-1) - [\mathrm{A}(\mathrm{N}) \cdot \mathrm{A}(\mathrm{N}-1)]\,\mathrm{A}(\mathrm{N})$$

et ensuite par $\dfrac{\mathrm{A}(\mathrm{N}-1)}{|\mathrm{A}(\mathrm{N}-1)|}$, | | désignant la norme, etc.

Il en résulte une première ligne A (0) orthogonale à toutes les autres. La solution du problème sera finalement :

$$x = \left(-\frac{a_{01}}{a_{00}}, -\frac{a_{02}}{a_{00}}, \dots, -\frac{a_{0n}}{a_{00}}\right)$$

Une valeur a_{00} très faible ou nulle indique un système incompatible où dét $\mathrm{A} = 0$.

Le programme répète le procédé d'orthogonalisation jusqu'à ce que le vecteur A (0) soit *suffisamment orthogonal* à tous les autres (paramètre P : précision du procédé de Schmidt).

Ensuite on détermine les résidus :

$$r = b - \mathrm{A}x$$

et on améliore la solution x en ajoutant celle du système :

$$\mathrm{A}x_1 = r$$

calculée de la même manière.

Le procédé est réitéré (variable II) jusqu'à ce que la moyenne quadratique des résidus soit assez près de zéro.

ORGANISATION DU PROGRAMME

Le cœur du programme est constitué par les lignes :

1100-1250	Algorithme de l'orthonormalisation.
1000-1060	Création de la matrice A à (N + 1) lignes et colonnes.
1070	NI = 0 : NI Compteur de nombre d'orthonormalisations.
1100-1250	Algorithme de Schmidt

G = N, N − 1, ..., 0 :

1. On calcule le produit scalaire KZ

$$KZ = \Sigma\, A(G, I) \cdot A(N, I)$$

2. PS = KZ : on orthogonalise les vecteurs

A (G, I) et A (R + 1, I) :

$$A(G, I) \longleftarrow A(G, I) - A(R + 1, I) \cdot PS$$

pour R = N − 1, ..., G, et on calcule le produit scalaire

$$KZ = \Sigma\, A(G, I) \cdot A(R, I)$$

3. On remplace A (G, I) par :

$$\frac{A(G, I)}{SQR(KZ)};$$

la norme de la $G^{ième}$ ligne est donc égale à 1.

1300-1440 On contrôle si A (0, I) est *suffisamment orthogonal* à toutes les autres lignes A (G, I).

(Les produits scalaires ne diffèrent de zéro que d'une valeur très faible.)

Remarquons finalement qu'il faut introduire l'ordre n du système : 30 INPUT N, la précision P et la matrice A du système $Ax = b$ ainsi que le vecteur b en lignes

10000 DATA... 10010 DATA..., etc.

PROGRAMME

```
 10 REM *** Solution d'un systeme ***
 15 REM *** lineaire carre  Ax = b ***
 20 REM *** d'apres M.EHRICH,M.OHSMANN *
**
 30 INPUT "L'ordre du systeme n = ";N
 40 INPUT "Precision du procede d'orthog
onalisation de SCHMIDT p = ";P
 50 DIM Z(N,N),A(N,N),B(N),X(N),SM(N)
 60 GOSUB 2010
 70 GOSUB 5010
 990 REM *** Sous-programmes ***
 1000 REM *** Orthonormalisation ***
 1010 FOR K  = 0 TO N
 1020 FOR I = 0 TO N
 1030 A(K,I) = Z(K,I)
 1040 NEXT
 1050 A(K,0) = SM(K): A(0,K) = SM(K)
 1060 NEXT
 1070 NI=0
 1100 FOR G = N TO 0 STEP -1
 1110 KZ = 0: M=0
 1120 FOR R = N TO G STEP -1
 1130 PS = KZ: KZ = 0: L=R+M: M=1
 1140 FOR I = 0 TO N
 1150 SS = A(G,I) - A(L,I)*PS
 1160 A(G,I) = SS: KZ = SS*A(R,I) + KZ
 1170 NEXT I
 1180 NEXT R
 1200 IF KZ = 0 THEN PRINT "Det A = 0? "
:END
 1210 PS = 1/SQR(KZ)
 1220 FOR I = 0 TO N
```

```
1230 A(G,I) = A(G,I)*PS
1240 NEXT I
1250 NEXT G
1300 PS=0
1310 FOR G=1 TO N
1320 KZ=0
1330 FOR I = 0 TO N
1340 KZ = A(0,I)*A(G,I) + KZ
1350 NEXT
1360 PS = ABS(KZ) + PS
1370 NEXT
1380 KZ = PS/N
1390 PRINT "Orthogonalite de la solutio
n: ";KZ
1400 NI= NI+1
1410 IF (KZ>P) AND (NI<4) THEN 1100
1420 IF KZ > P THEN PRINT "Precision ";
P;" non obtenue "
1430 IF ABS(A(0,0)) < 1E-10 THEN PRINT
"Systeme incompatible ?? ":PRINT
1440 IF   A(0,0) = 0 THEN END
1500 S=-1/A(0,0)
1510 REM *** Solution ***
1520 FOR K=1 TO N
1530 A(0,K) = A(0,K)*S
1540 NEXT
1550 RETURN
2000 REM *** Initialisation-Introductio
n de la matrice A ***
2010 FOR K=1 TO N
2020 FOR I = 1 TO N
2030 READ Z(K,I)
2040 NEXT
2050 NEXT
2060 FOR K = 1 TO N
2070 READ B(K)
2080 NEXT
2090 PRINT " Systeme d'equations lineai
res:"
2100 FOR K = 1 TO N
2110 PRINT
2120 FOR I = 1 TO N
```

```
 2130 PRINT Z(K,I);" ";
 2140 NEXT
 2150 PRINT "  ... ";B(K)
 2160 NEXT
 2170 PRINT
 2180 RETURN
 5000 REM *** Iteration -SECOND MEMBRE *
**
 5010 II = 0
 5020 FOR K = 1 TO N
 5030 SM(K) = B(K)
 5040 X(K) = 0
 5050 NEXT
 5100 REM *** Execution d'un pas ***
 5110 GOSUB 1010
 5120 PRINT " Solution apres ";II;" iter
ations: "
 5130 FOR K = 1 TO N
 5140 X(K) = X(K) + A(0,K)
 5150 PRINT X(K)
 5160 NEXT
 5200 REM ** Calcul des residus  ***
 5210 PRINT " Residus: "
 5220 S1 = 0
 5300 FOR K = 1 TO N
 5310 S = B(K)
 5320 FOR I = 1 TO N
 5330 S = S - Z(K,I)*X(I)
 5340 NEXT
 5350 SM(K) = S
 5360 PRINT S
 5370 S1 = S1 + S*S
 5380 NEXT
 5390 PRINT " Moyenne quadratique = ";S1
/N:IF S1/N<1E-12 THEN END
 5400 II = II + 1
 5410 IF II > 3 AND S1/N > 1E-5 THEN PRI
NT " Systeme incompatible?? ":END
 5420 GOTO 5110
 9990 REM *** Matrice A par ligne ***
 10000 DATA -74,80,18,-11,-4,-8
 10010 DATA 14,-69,21,28,0,7
```

```
10020 DATA 66,-72,-5,7,1,4
10050 DATA -12,66,-30,-23,3,-3
10060 DATA 3,8,-7,-4,1,0
10070 DATA 4,-12,4,4,0,1
10090 REM *** Second membre b ***
10100 DATA 1,0,0,0,0,0
```

MEILLEURE APPROXIMATION EN MOYENNE

Partons d'un support d'approximation (t_1, ..., t_n) (ce sont les points où la fonction est parfaitement déterminée) et d'un ensemble de points (t_m, y_m), $m=1$, ..., n.

Nous voulons trouver un polynôme p tel que la somme :

$$\sum_{m=1}^{n} (y_m - p(t_m))^2, \quad p(t) = \sum_{k=1}^{n} x_k t^{k-1}$$

soit minimale.

Pour trouver les coefficients x_k, il faut résoudre le système d'équations linéaires :

$$\mathrm{A}x = b$$

avec :

$$a_{ki} = \sum_{m=1}^{n} t_m^{k+i-2}, \quad b_k = \sum_{m=1}^{n} y_m t_m^{k-1}$$

Nous modifions donc ainsi le programme « Résolution d'un système linéaire... » :

```
 2000 REM *** Initialisation-Introductio
n de la matrice A ***
 2005 DIM T(N),Y(N),S(N)
```

```
2006 PRINT:PRINT "Points t(m), y(m): "
2010 FOR M=1 TO N
2015 READ T(M),Y(M):PRINTM,T(M),Y(M)
2020 S(M)=1:NEXT
2025 FOR K=1 TO N
2030 S=0:T=0
2035 FOR M=1 TO N
2040 S=S+S(M):T=T+Y(M)
2045 S(M)=S(M)*T(M):Y(M)=Y(M)*T(M)
2050 NEXT
2055 Z(K,1)=S:B(K)=T
2060 NEXT
2065 FOR I=2 TO N
2070 S=S(1):S(1)=S(1)*T(1)
2075 FOR K=1 TO N-1
2080 Z(K,I)=Z(K+1,I-1):S=S+S(K+1)
2085 S(K+1)=S(K+1)*T(K+1)
2090 NEXT K:Z(N,I)=S
2095 NEXT I
2099 PRINT " Systeme d'equations lineai
res:"
2100 FOR K = 1 TO N

9990 REM *** Points t(m), y(m) ***
10000 DATA 0,0
10010 DATA .5235987756,.5
10020 DATA 1.570796327,1
10030 DATA 2.617993878,.5
10040 DATA 3.141592654,0
```

Résultat :

```
 Solution apres 0  iterations:
x(1 ) = 4.7E-08
x(2 ) = .986760139
x(3 ) = .050661285
x(4 ) = -.232211355
x(5 ) = .03695758
```

MATRICE TRIDIAGONALE

Initiation au problème

La solution d'une équation différentielle du second ordre dépend de deux constantes arbitraires. Pour arriver à une solution unique on impose donc deux conditions supplémentaires en forme de conditions initiales ou de conditions aux limites, une à chaque extrémité de l'intervalle.

Considérons le problème suivant :

$$\left.\begin{aligned} &-y'' + q(x)\,y = g(x), \quad a \leqslant x \leqslant b \\ &y(a) = \alpha, \quad y(b) = \beta \end{aligned}\right| \quad y = ?$$

y, q, g étant des fonctions à valeurs complexes.

La méthode la plus simple pour attaquer ce problème est celle des différences finies (voir le Chapitre 16).

On partage l'intervalle $[a, b]$ en $n+1$ intervalles égaux de longueur :

$$h = \frac{(b-a)}{(n+1)}$$

et on pose :

$$a = x_0 < x_1 < \ldots < x_k = a + kh < \ldots < x_{n+1} = b$$

$$y_k = y(x_k), \quad q_k = q(x_k), \quad g_k = g(x_k).$$

En remplaçant $y''(x_k)$ par la valeur approchée :

$$D^2\, y_k = \frac{y_{k+1} - 2\,y_k + y_{k-1}}{h^2}, \quad k = 1, \ldots, n$$

on obtient finalement un système d'équations linéaires dont la matrice est tridiagonale :

$$y_0 = \alpha$$

$$\text{conditions aux limites} \quad \frac{2\,y_k - y_{k-1} - y_{k+1}}{h^2} + q_k\,y_k = g_k, \quad k = 1, \ldots, n$$

$$y_{n+1} = \beta$$

(ou sous forme matricielle) :

$$\frac{1}{h^2}\begin{pmatrix} 2+q_1h^2 & -1 & 0 & 0 & . & . & 0 \\ -1 & 2+q_2h^2 & -1 & 0 & . & . & 0 \\ 0 & -1 & 2+q_3h^2 & -1 & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & -1 & 2+q_{n-1}h^2 & -1 \\ 0 & 0 & 0 & . & 0 & -1 & 2+q_nh^2 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ . \\ . \\ y_{n-1} \\ y_n \end{pmatrix}=\begin{pmatrix} g_1+\dfrac{\alpha}{h^2} \\ g_2 \\ g_3 \\ \\ . \\ g_{n-1} \\ g_n+\dfrac{\beta}{h^2} \end{pmatrix}$$

Remarque

Une matrice tridiagonale possède des éléments non nuls uniquement dans la diagonale principale et les deux diagonales adjacentes.

Méthode

Partant de l'équation $Ax=a$, A tridiagonale on met la matrice A sous la forme :

$$A=G\cdot D$$

où G est une matrice triangulaire inférieure très creuse (où la presque totalité des éléments est nulle) et D une matrice triangulaire supérieure :

$$A=\begin{pmatrix} d_1 & s_1 & 0 & 0 & . & . & 0 \\ i_2 & d_2 & s_2 & 0 & . & . & 0 \\ 0 & i_3 & d_3 & s_3 & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & . & . & 0 & i_{n-1} & d_{n-1} & s_{n-1} \\ 0 & 0 & . & . & 0 & i_n & d_n \end{pmatrix}$$

$$G = \begin{pmatrix} d_1^* & 0 & 0 & 0 & . & . & 0 \\ i_2 & d_2^* & 0 & 0 & . & . & 0 \\ 0 & i_3 & d_3^* & 0 & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & . & . & 0 & i_n & d_n^* \end{pmatrix}$$

$$D = \begin{pmatrix} 1 & s_1^* & 0 & 0 & . & . & 0 \\ 0 & 1 & s_2^* & 0 & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & . & . & . & 0 & 1 & s_{n-1}^* \\ 0 & . & . & . & 0 & 0 & 1 \end{pmatrix}$$

La résolution des deux systèmes :

$$Gi^* = a \quad \text{et} \quad Dx = i^*$$

est immédiate. Notons alors l'algorithme :

$$d_1^* = d_1$$

$$i_1^* = \frac{a_1}{d_1^*}$$

$k = 2, \ldots, n$: $\quad s_{k-1}^* = \dfrac{s_{k-1}}{d_{k-1}^*}$

$$d_k^* = d_k - i_k s_{k-1}^*$$

$$i_k^* = \frac{(a_k - i_k i_{k-1}^*)}{d_k^*}$$

$$x_n = i_n^*$$

$k = n-1, \ldots, 1$: $\quad x_k = i_k^* - s_k^* x_{k+1}$

Remarque

Une matrice triangulaire supérieure (inférieure) possède uniquement des éléments nuls en dessous (au-dessus) de la diagonale.

Exemple numérique et programme

```
Ordre de la matrice n = 3
 Diagonale Re d(1 ),Im d(1 ) = 1  1
 Diagonale Re d(2 ),Im d(2 ) = 1  1
 Diagonale Re d(3 ),Im d(3 ) = 3  -1
 Surdiagonale Re s(1 ),Im s(1 ) = 2  1
 Surdiaonale Re s(2 ),Im s(2 ) = 1  -1
 Sousdiagonale Re i(2 ),Im i(2 ) = 1  -.5
 Sousdiagonale Re i(3 ),Im i(3 ) = 0  1
 Second membre Re a(1 ),Im a(1 ) = 3  3
 Second membre Re a(2 ),Im a(2 ) = 4.5  3
 Second membre Re a(3 ),Im a(3 ) = 4  4
 Solution x(1 ) = 0 +j*1
 Solution x(2 ) = 2 +j*0
 Solution x(3 ) = 1 +j*1
```

```
 10 REM *** Systeme d'equations ***
 20 REM *** lineaires - matrice ***
 30 REM *** tridiagonale        ***
 40 REM *** coefficients complexes ***
 90 REM *** Introduction des coefficient
s - Partie reelle, P. imaginaire ***
 100 INPUT "Ordre de la matrice n = ";N
 110 DIM RD(N),ID(N), RS(N-1), IS(N-1),R
I(N), II(N), RA(N), IA(N)
 120 FOR K =1 TO N
 130 PRINT" Diagonale Re d(";K;"),Im d("
;K;") = ";:INPUT RD(K),ID(K)
 140 NEXT
 150 FOR K = 1 TO N-1
 160 PRINT" Surdiagonale Re s(";K;"),Im
s(";K;") = ";:INPUT RS(K),IS(K)
```

```
 170 NEXT
 180 FOR K=2 TO N
 190 PRINT" Sousdiagonale Re i(";K;"),Im
 i(";K;") = ";:INPUTRI(K),II(K)
 200 NEXT
 210 FOR K=1 TO N
 220 PRINT" Second membre Re a(";K;"),Im
 a(";K;") = ";:INPUTRA(K),IA(K)
 230 NEXT
 290 REM *** Algorithme ***
 300 S=RD(1)*RD(1)+ID(1)*ID(1)
 310 RI(1)=(RA(1)*RD(1)+IA(1)*ID(1))/S
 320 II(1)=(IA(1)*RD(1)-RA(1)*ID(1))/S
 330 FOR K=2 TO N
 340 RS=(RS(K-1)*RD(K-1)+IS(K-1)*ID(K-1)
)/S
 350 IS(K-1)=(IS(K-1)*RD(K-1)-RS(K-1)*ID
(K-1))/S:RS(K-1)=RS
 360 RD(K)=RD(K)-RI(K)*RS(K-1)+II(K)*IS(
K-1)
 370 ID(K)=ID(K)-II(K)*RS(K-1)-RI(K)*IS(
K-1)
 380 S=RD(K)*RD(K)+ID(K)*ID(K)
 390 RI=(RA(K)-RI(K)*RI(K-1)+II(K)*II(K-
1))/S
 400 II=(IA(K)-RI(K)*II(K-1)-II(K)*RI(K-
1))/S
 410 RI(K)=RI*RD(K)+II*ID(K):II(K)=II*RD
(K)-RI*ID(K)
 420 NEXT
 430 FOR K=N-1 TO 1 STEP -1
 440 RI(K) = RI(K)-RS(K)*RI(K+1)+IS(K)*I
I(K+1)
 450 II(K) = II(K)-RS(K)*II(K+1)-IS(K)*R
I(K+1)
 460 NEXT
 490 REM *** Resultats ***
 500 FOR K = 1 TO N
 510 PRINT " Solution x(";K;") = "; RI(K
);"+j*";II(K)
 520 NEXT
 530 END
```

AUTRES EXEMPLES NUMÉRIQUES

Exemple 1

```
L'ordre du systeme n = ? 6
Precision du Procede d'orthogonalisation
 de SCHMIDT p = ? 1E-9
 Systeme d'equations lineaires:

-74   80   18  -11   -4   -8      ...   1

 14  -69   21   28    0    7      ...   0

 66  -72   -5    7    1    4      ...   0

-12   66  -30  -23    3   -3      ...   0

 3    8   -7   -4    1    0      ...   0

 4  -12    4    4    0    1      ...   0

Orthogonalite de la solution:
 1.68930146E-05
Orthogonalite de la solution:
 3.45607987E-11
 Solution apres  0  iterations:
 1.00000093
-7.4043382E-07
-2.00000345
 15.0000231
 43.0000715
-56.0000913
 Residus:
-1.1920929E-07
 1.78813934E-07
 5.96046448E-08
-1.1920929E-07
-1.49011612E-08
 1.49011612E-08
 Moyenne quadratique =  1.07321559E-14
```

Exemple 2

```
L'ordre du systeme n = ? 6
Precision du Procede d'orthogonalisation
 de SCHMIDT p = ? 1E-9
 Systeme d'equations lineaires:

-74   80   18  -11  -4  -8     ...  0

 14  -69   21   28   0   7     ...  0

 66  -72  -5    7    1   4     ...  0

-12   66  -30  -23   3  -3     ...  1

 3    8  -7   -4    1   0     ...  0

 4  -12   4    4    0   1     ...  0

Orthogonalite de la solution:
 5.6354717E-08
Orthogonalite de la solution:
 4.85063841E-11
 Solution apres  0  iterations:
-40.0001156
 35.0000928
 155.000431
-1034.00289
-3211.00892
 4096.0114
 Residus:
 0
 3.81469727E-06
 7.62939453E-06
 0
 0
 0
 Moyenne quadratique =  1.2126596E-11
```

```
Orthogonalite de la solution:
 4.62746909E-03
Orthogonalite de la solution:
 1.57266792E-12
 Solution apres  1  iterations:
-40.000169
 35.00015
 155.00066
-1034.0044
-3211.01366
 4096.01742
 Residus:
 0
 3.81469727E-06
 0
 0
 0
 0
 Moyenne quadratique =  2.42531921E-12
Orthogonalite de la solution:
 .061984806
Orthogonalite de la solution:
 1.81756832E-11
 Solution apres  2  iterations:
-40.000169
 35.0001538
 155.000667
-1034.00444
-3211.01382
 4096.01762
 Residus:
 0
-3.81469727E-06
 0
 0
 0
 0
 Moyenne quadratique =  2.42531921E-12
Orthogonalite de la solution:
 .061984806
```

```
Orthogonalite de la solution:
 1.81756832E-11
 Solution apres  3  iterations:
-40.000169
 35.00015
 155.00066
-1034.0044
-3211.01366
 4096.01742
 Residus:
 0
 3.81469727E-06
 0
 0
 0
 0
 Moyenne quadratique =  2.42531921E-12
```

```
Orthogonalite de la solution:
 .061984806
Orthogonalite de la solution:
 1.81756832E-11
 Solution apres  4  iterations:
-40.000169
 35.0001538
 155.000667
-1034.00444
-3211.01382
 4096.01762
 Residus:
 0
-3.81469727E-06
 0
 0
 0
 0
 Moyenne quadratique =  2.42531921E-12
```

```
Systeme mal conditionne...
Solution exacte calculee par le programme FADEEV:
-40
35
155
-1034
-3211
4096
```

Exemple 3

```
L'ordre du systeme n = ? 3
Precision du procede d'orthogonalisation
 de SCHMIDT p = ? 1E-9
 Systeme d'equations lineaires:

 1    2    3     ...   6

 2    3    4     ...   9

 3    4    5     ...   12

 det A = 0

Orthogonalite de la solution:  .4102592
Orthogonalite de la solution:
 9.11920022E-10
 Solution apres  0  iterations:
-6.97340143
 16.9468027
-6.97340142
 Residus:
 2.57045031E-07
 3.87430191E-07
 5.25265932E-07
 Moyenne quadratique =  1.640262E-13
```

Exemple 4

```
L'ordre du systeme n = ? 2
Precision du procede d'orthogonalisation
 de SCHMIDT p = ? 1E-9
 Systeme d'equations lineaires:

 1   1     ...  1

 1   1     ...  2

Orthogonalite de la solution:
 2.56113708E-09
Orthogonalite de la solution:
 5.82076609E-11
Systeme incompatible ??

 Solution apres  0  iterations:
 8.98496656E+09
-8.98496656E+09
 Residus:
 0
 0

 Moyenne quadratique = 0
```

UNE MÉTHODE ITÉRATIVE

Parmi les nombreuses méthodes itératives (où l'on applique le même procédé plusieurs fois; voir le Chapitre 16), nous proposons celle de Gauss-Seidel et une généralisation que nous allons utiliser dans le Chapitre « Équations différentielles elliptiques ».

Comme le programme est largement commenté, nous donnons tout de suite son listing.

Programme

```
 10 REM SOLUTION PAR ITERATION
 20 REM METHODE ADAPTEE AUX GRANDS
 30 REM SYSTEMES LINEAIRES CREUX
 40 REM AX=B, A POSSEDANT UNE
 50 REM DIAGONALE PRINCIPALE DOMINANTE
 60 REM |A(I,I)|> SOMME(|A(I,K)|,K<>I)
 70 REM PARAMETRE W DE LA METHODE SOR
 80 REM (SUCCESSIVE OVER (OR UNDER)
 90 REM RELAXATION, 0<W<2)
 100 INPUT "Ordre de la matrice A: N= ";
N
 110 INPUT "Parametre W= ";W
 120 INPUT "Precision du resultat P= ";P
 130 INPUT "Nombre d'iterations NI= ";NI
: NJ=NI
 140 DIM A(N,N), B(N), X(N)
 190 REM INITIALISATION
 200 FOR I=1 TO N
 210 FOR K=1 TO N
 220 READ A(I,K)
 230 NEXT:NEXT
 240 FOR I=1 TO N
 250 READ B(I): X(I)=0: REM START
 260 NEXT
 270 FOR I=1 TO N
 280 PRINT: FOR K=1 TO N
 290 PRINT A(I,K),
 300 NEXT
 310 PRINT "..." B(I)
 320 NEXT
 390 REM ALGORITHME
 400 NJ=NJ-1
 410 R=0: REM RESIDU
 420 FOR I=1 TO N
 430 SS=0
 440 FOR K=1 TO N
 450 SS=SS+ A(I,K)*X(K)
 460 NEXT
 470 X=X(I)
```

```
 480 X(I)=X-W*(SS-B(I))/A(I,I)
 490 R=R+ABS(X-X(I))
 500 NEXT
 510 IF R<P THEN 540
 520 IF NJ>0 THEN 400
 530 PRINT:PRINT "Precision ";P;"non obt
enue apres ";NI;" iterations."
 535 PRINT "Solution X(I) approchee:":GO
TO 550
 540 PRINT:PRINT"Solution X(I), I=1,....,
N, apres ";NI-NJ;"iterations:"
 550 FOR I=1 TO N
 560 PRINT "X(";I;") = ";INT(X(I)/P+.5)*
P
 570 NEXT
 580 END
 590 REM MATRICE A PAR LIGNE
 600 DATA 4,-1,0,-1
 610 DATA -1,4,-1,0
 620 DATA 0,-1,4,-1
 630 DATA -1,0,-1,4
 690 REM B(I), SECOND MEMBRE
 700 DATA 1,0,0,0
```

IV
INTERPOLATION

13. LISSAGE PAR DES FONCTIONS DE SPLINE

INITIATION AU PROBLÈME

Lorsqu'une fonction y est définie par un procédé expérimental ou numérique — on relève par exemple des valeurs à certains moments x_k ou on résoud une équation différentielle par une méthode à pas —, on aimerait compléter cette fonction pour tout x par une interpolation.

Citons un exemple et considérons un ensemble de températures relevées à Marseille un jour de septembre, la coordonnée x représentant l'heure du jour, la coordonnée y la température relevée.

xh	y °C
6	16,5
8	18,9
10	23,8
12	28,4
14	29,6
16	28,7
18	23,1
20	19,4
22	17,3

On cherche la courbe interpolante et la température à 14 h 30 et à 21 h.

MÉTHODE

Une certaine incertitude sur les valeurs de la fonction aux points d'interpolation interdit l'interpolation par des polynômes de haut degré. Nous employons donc des polynômes p_k de troisième degré pour interpoler la fonction inconnue entre deux points du support d'interpolation.

$$(x_k, y(x_k)) \text{ et } (x_{k+1}, y(x_{k+1}))$$

$$p_k(x) = a_k + b_k(x - x_k) + c_k(x - x_k)^2 + d_k(x - x_k)^3$$

$$k = 0, \ldots, n-1$$

Les polynômes seront définis par les relations :

$$p_0(x_0) = y(x_0)$$

$$p_k(x_{k+1}) = p_{k+1}(x_{k+1}) = y(x_{k+1})$$

$$\frac{dp_k}{dx}(x_{k+1}) = \frac{dp_{k+1}}{dx}(x_{k+1}) \qquad k = 0, 1, \ldots, n-2$$

$$\frac{d^2 p_k}{dx^2}(x_{k+1}) = \frac{d^2 p_{k+1}}{dx^2}(x_{k+1})$$

$$p_{n-1}(x_n) = y(x_n)$$

Par conséquent, on arrive à une interpolation polynomiale par différents polynômes de bas degré en posant :

$$p(x) = p_k(x), \; x_k \leqq x \leqq x_{k+1}, \; k = 0, \ldots, n-1$$

p et ses deux dérivées existent et sont continues.

Comme les polynômes p_k ne sont pas encore déterminés d'une façon unique par les relations indiquées ci-dessus, on peut introduire une des trois conditions supplémentaires suivantes :

1. $p_0''(x_0) = p_{n-1}''(x_n) = 0$

2. $p_0^{(k)}(x_0) = p_{n-1}^{(k)}(x_n)$, $k = 0, 1, 2$ périodicité

3. $p_0'(x_0) = a$, $p_{n-1}'(x_n) = b$

Nous nous bornons au premier cas et nous voulons calculer les coefficients des polynômes p_k.

Nos conditions nous mènent à un système d'équations linéaires :

$$\begin{pmatrix} f_1 & h_1 & 0 & 0 & . & . & . & 0 \\ h_1 & f_2 & h_2 & 0 & . & . & . & 0 \\ 0 & h_2 & f_3 & h_3 & . & . & . & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & 0 & h_{n-3} & f_{n-2} & h_{n-2} \\ 0 & 0 & 0 & . & . & 0 & h_{n-2} & f_{n-1} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ . \\ . \\ c_{n-2} \\ c_{n-1} \end{pmatrix} = \begin{pmatrix} g_1 \\ g_2 \\ g_3 \\ . \\ . \\ g_{n-2} \\ g_{n-1} \end{pmatrix}$$

avec :

$$f_k = 2\,(h_k + h_{k-1}),\ h_k = x_{k+1} - x_k,\ y_k = y\,(x_k)$$

$$g_k = \frac{3\,(y_{k+1} - y_k)}{h_k} - \frac{3\,(y_k - y_{k-1})}{h_{k-1}}$$

Heureusement sa matrice est triagonale, ce qui facilite la résolution (voir le paragraphe : « Résolution d'un système d'équations linéaires »).

Le calcul explicite des autres coefficients ne présente aucune difficulté :

$$a_k = y\,(x_k) = y_k$$

$$a_n := y_n \qquad c_0 = c_n = 0$$

$$b_k = \frac{a_{k+1} - a_k}{h_k} - \frac{h_k\,(c_{k+1} + 2\,c_k)}{3}$$

$$d_k = \frac{(c_{k+1} - c_k)}{(3\,h_k)}$$

$$k = 0, \ldots, n-1$$

DÉROULEMENT DU PROGRAMME

Initialisation 10-180	Il faut fournir au programme lorsqu'il le demande le nombre N et les points d'interpolation $x(k)$, $y(k)$; $k=0, ..., N$, $x(k) \leqslant x(k+1)$
Algorithme 200-410	Le programme calcule les coefficients des polynômes : $p_k = a_k + b_k(x-x_k) + c_k(x-x_k)^2 + d_k(x-x_k)^3$
Résultats 420-570	Affichage des coefficients sur l'écran Calcul des coordonnées d'un ou plusieurs points interpolés
Partie graphique 580-750	Dessin du polynôme interpolant p en haute résolution

LE PROGRAMME

```
 10 REM *** Lissage par des fonctions sp
line ***
 20 PRINT"Introduction des points d'inte
rpolation x(0),y(0) x(1),y(1)...x(n),y(n
)"
 30 INPUT "n = (n>=3)";N
 40 DIMA(N),B(N-1),C(N),D(N-1),E(N-1),F(
N-1),G(N-1),H(N-1),X(N),Y(N)
 50 FOR K=0 TO N
 60 PRINT "Point # ";K;" x,y = ";
 70 INPUT X(K),Y(K)
 80 IF K>0 THEN IF X(K)<= X(K-1) THEN PR
INT"Choisissez x(k)>x(k-1) ":GOTO 70
 90 A(K)=Y(K): NEXT
```

```
 100 PRINT "Toutes les valeurs justes? O
/N ": GET T$
 110 IF T$<> "N" THEN 200
 120 PRINT "Quel point voulez-vous corri
ger? entrez k ": GET T$
 130 K=VAL(T$):PRINT "Point # ";K;" x,y
= ";
 140 INPUT X(K),Y(K)
 150 IF K>0 THEN IF X(K)<= X(K-1) THEN P
RINT"Choisissez x(k)>x(k-1) ":GOTO 140
 160 IF K<N THEN IF X(K)>= X(K+1) THEN P
RINT"Choisissez x(k)<x(k+1) ":GOTO 140
 170 A(K)=Y(K)
 180 GOTO 100
 190 REM *** Matrice triagonale ***
 200 FOR K=0 TO N-1
 210 H(K) = X(K+1) - X(K)
 220 NEXT
 230 FOR K=1 TO N-1
 240 F(K) = 2*(H(K)+H(K-1)):G(K)=3*(A(K+
1)-A(K))/H(K)-3*(A(K)-A(K-1))/H(K-1)
 250 NEXT
 260 REM *** Resolution du systeme linea
ire ***
 270 C(1) = G(1)/F(1)
 280 FOR K=2 TO N-1
 290 E(K-1) = H(K-1)/F(K-1)
 300 F(K)  = F(K) - H(K-1)*E(K-1)
 310 C(K) =(G(K) - H(K-1)*C(K-1))/F(K)
 320 NEXT
 330 FOR K = N-2 TO 1 STEP -1
 340 C(K) = C(K) - E(K)*C(K+1)
 350 NEXT
 360 REM *** Coefficients des polynomes
***
 370 C(0) = 0: C(N) = 0
 380 FOR K=0 TO N-1
 390 B(K) = (A(K+1)-A(K))/H(K) -(C(K+1)
+ 2*C(K))*H(K)/3
 400 D(K) = (C(K+1) - C(K))/(3*H(K))
 410 NEXT
```

```
 420 REM *** Affichage des coefficients
***
 430 PRINT"Polynomes Pk(x), k=0,...,n-1
:"
 440 PRINT"ak+bk(x-xk )+ck(x-xk )^2+dk(x-x
k )^3 "
 450 FOR K = 0 TO N-1
 460 PRINT "k = ";K;" x(k ) = ";X(K);" x(
k+1 ) = ";X(K+1 )
 470 PRINT A(K),B(K),C(K),D(K)
 480 NEXT
 490 PRINT:PRINT   " Calcul des coordonn
ees d'un Point interPole "
 500 INPUT "Valeur de la variable x = ";
X
 510 IF X<X(0) OR X> X(N) THEN PRINT "x
en dehors de l'intervalle ":GOTO 500
 520 K=0
 530 IF X>X(K+1 ) THEN K=K+1: GOTO 530
 535 X=X-X(K)
 540 PRINT"P(";X+X(K);") = ";((D(K)*X+C(
K))*X+B(K))*X +A(K)
 550 PRINT"D'autres Points ? O/N ": GET
T$
 560 IF T$<> "N" THEN 500
 570 REM *** Courbe obtenue Par interPol
ation ***
 580 PRINT "Dessin sur l'ecran ? O/N ":
GET T$: IF T$="N" THEN END
 590 INPUT "Ordonnees extremes y min, y
max ";Y1,Y2
 600 IF Y1 >= Y2 THEN PRINT " y min < y
max ! ": GOTO 590
 610 HIRES:REM Ecran Haute resolution
 620 K = 0
 630 FOR I = 0 TO 239
 640 X = X(0) + (X(N)-X(0))*I/239
 650 IF X>X(K+1 ) AND K<N-1   THEN K=K+1:
 GOTO 650
 655 X= X-X(K)
 660 Y = ((D(K)*X + C(K))*X + B(K))*X +
A(K)
```

```
 670 Y = INT ((Y2-Y)*191/(Y2 -Y1)+.5)
 680 IF Y<0 THEN Y=0
 690 IF Y>191 THEN Y=191
 700 CURSET I,Y,1
 710 NEXT
 720 CURSET 0,0,1
 730 DRAW 0,191,1
 740 DRAW 239,0,1
 750 POKE26,96:REMPermet d'ecrire dans l
es 3 lignes en bas de l'ecran Haute Res
 770 PRINT"Enfoncer une touche pour cont
inuer";
 780 GET A$
 800 TEXT
 810 END
```

QUELQUES EXEMPLES NUMÉRIQUES

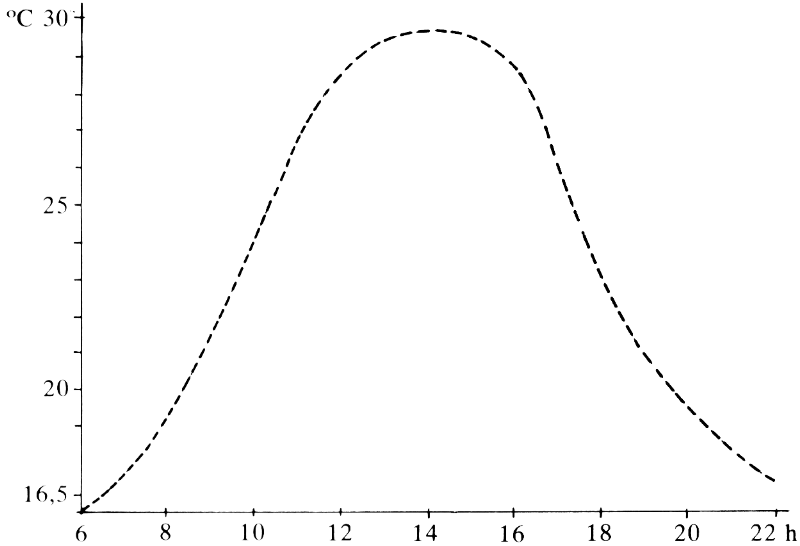
Courbe de températures

```
Introduction des points d'interpolatio
n x(0),y(0) x(1),y(1)...x(n),y(n)
n = (n>=3)? 8
Point #  0  x,y = ? 06,16.5
Point #  1  x,y = ? 08,18.9
Point #  2  x,y = ? 10,23.8
Point #  3  x,y = ? 12,28.4
Point #  4  x,y = ? 14,29.6
Point #  5  x,y = ? 16,28
Point #  6  x,y = ? 18,23.1
Point #  7  x,y = ? 20,19.4
Point #  8  x,y = ? 22,17.3
Toutes les valeurs justes? O/N
Quel point voulez-vous corriger? entrez
k
Point #  5  x,y = ?
? 16,28.7
Toutes les valeurs justes? O/N
```

```
Polynomes Pk(x), k=0,...,n-1 :
ak+bk(x-xk)+ck(x-xk)^2+dk(x-xk)^3
k =  0  x(k) =  6  x(k+1) =  8
 16.5    .885106774      0
 .0787233063
k =  1  x(k) =  8  x(k+1) =  10
 18.9    1.82978645      .472339838
-.0811165315
k =  2  x(k) =  10  x(k+1) =  12
 23.8    2.74574742     -.0143593515
-.10425718
k =  3  x(k) =  12  x(k+1) =  14
 28.4    1.43722386     -.63990243
 .11064525
k =  4  x(k) =  14  x(k+1) =  16
 29.6    .205357144      .0239690719
-.175823822
k =  5  x(k) =  16  x(k+1) =  18
 28.7   -1.80865243     -1.03097386
 .267650037
k =  6  x(k) =  18  x(k+1) =  20
 23.1   -2.72074743      .574926362
-.0697763254
k =  7  x(k) =  20  x(k+1) =  22
 19.4   -1.25835788      .15626841
-.026044735

 Calcul des coordonnees d'un point inter
pole
Valeur de la variable x = ?
? 14.5
P( 14.5 ) =  29.6866929
D'autres points ? O/N
Valeur de la variable x = ?
? 21
P( 21 ) =  18.2718658
D'autres points ? O/N
Dessin sur l'ecran ? O/N


Ordonnees extremes y min, y max: 16.5, 30
```




```
Introduction des Points d'interPolatio
n x(0),y(0) x(1),y(1)...x(n),y(n)
n = (n>=3)? 3
Point #  0  x,y = ? 0,0
Point #  1  x,y = ? 1,2
Point #  2  x,y = ? 2,1
Point #  3  x,y = ? 3,4
Toutes les valeurs Justes? O/N
Polynomes Pk(x), k=0,...,n-1 :
ak+bk(x-xk)+ck(x-xk)^2+dk(x-xk)^3
k =  0  x(k) =  0  x(k+1) =  1
 0        3.06666667       0
-1.06666667
k =  1  x(k) =  1  x(k+1) =  2
 2        -.133333333      -3.2
 2.33333333
k =  2  x(k) =  2  x(k+1) =  3
 1        .466666667       3.8
-1.26666667
```

```
 Calcul des coordonnees d'un point inter
pole
Valeur de la variable x = ?
? 3
P( 3 ) =  4
D'autres points ? O/N
Valeur de la variable x = ?
? .5
P( .5 ) =  1.4
D'autres points ? O/N
Valeur de la variable x = ?
? 2.8
P( 2.8 ) =  3.1568
D'autres points ? O/N
Dessin sur l'ecran ? O/N
Ordonnees extremes y min, y max: 0, 4
```

14. INTERPOLATION ET TRANSFORMÉE DE FOURIER RAPIDE (FFT)

INITIATION AU PROBLÈME

Les séries de Fourier ainsi que la transformée de Fourier sont depuis longtemps des outils très puissants pour traiter les fonctions périodiques et non périodiques qui décrivent les phénomènes de la physique et de l'électronique. Citons l'amplitude d'un son en fonction du temps ou plus généralement le traitement de signaux et le calcul des spectres.

Le théorème de Fourier dit qu'à toute fonction g intégrable et périodique peut se substituer une série de fonctions sinusoïdales dont les pulsations sont multiples de la pulsation fondamentale :

$$\omega = 2\,\pi f = \frac{2\,\pi}{T}$$

$$g(t) \cong \frac{A_0}{2} + \sum_{n=1}^{\infty} (A_n \cos n\omega t + B_n \sin n\omega t)$$

Les coefficients sont des intégrales définies :

$$A_n = \frac{2}{T}\int_0^T g(t) \cos n\omega t dt,\; B_n = \frac{2}{T}\int_0^T g(t) \sin n\omega t dt$$

et $\frac{A_0}{2}$ désigne la valeur moyenne de la fonction g. Les fréquences nf et les coefficients A_n, B_n caractérisent donc le spectre de la fonction g (voir la Figure 14.1).

Dans le cas de fonctions non périodiques, la série de Fourier est remplacée par la transformée de même nom :

$$S(f) \cong \int_{-\infty}^{\infty} g(t)\, e^{-2\pi j f t}\, dt$$

et le spectre perd son caractère discontinu.

On peut reconstituer la fonction g de son spectre $S(f)$ par la transformée inverse :

$$g(t) = \int_{-\infty}^{\infty} S(f)\, e^{2\pi j f t}\, df$$

(si g est continue et si les intégrales :

$$\int_{-\infty}^{\infty} |g|\, dt, \int_{-\infty}^{\infty} |S|\, df$$

existent).

Comme les intégrales sont généralement difficiles à évaluer et que souvent on ne connaît de la fonction g que quelques valeurs, on les remplace par des sommes en nombre fini de la forme :

$$S(k) = \sum_{\ell=0}^{N-1} g_\ell\, e^{-2\pi j \ell k/N},\ k = 0, 1, \ldots, N-1$$

MÉTHODE

Soit f une fonction périodique de période 2π.
Prenons une base d'interpolation de fonctions trigonométriques :

$$1, \cos x, \sin x, \cos 2x, \sin 2x, \ldots, \cos nx, \sin nx, \ldots$$

Nous voulons interpoler la fonction f, dont on ne connaît les valeurs que dans les points x_k, par l'expression trigonométrique :

$$p(x)=\frac{A_0}{2}+\sum_{k=1}^{M} A_k \cos kx+B_k \sin kx, \ N=2M+1$$

$$p(x)=\frac{A_0}{2}+\sum_{k=1}^{M-1} A_k \cos kx+B_k \sin kx+\frac{A_M}{2}\cos Mx, \ N=2M$$

et nous admettons que les points d'interpolation :

$$(x_k, f(x_k)), \ k=0, ..., N-1$$

sont équipartis sur $[0,2\pi]$:

$$x_k=\frac{2\pi k}{N}, \ k=0, ..., N-1$$

On peut considérablement simplifier le problème en posant :

$$e^{kjx}=\cos kx+j \sin kx \text{ (formule de Moivre)}$$

et :

$$p(x)=b_0+b_1 e^{jx}+b_2 e^{2jx}+...+b_{N-1} e^{(N-1)jx}$$

p polynôme de phase, b_k coefficient complexe.

On vérifie facilement :

$$b_0=\frac{A_0}{2}, \ b_k=\frac{(A_k-jB_k)}{2}, \ b_{N-k}=\frac{(A_k+jB_k)}{2}$$

$$A_0=2b_0, \ A_k=b_k+b_{N-k}, \ B_k=j(b_k-b_{N-k}), \ k=1, ..., M$$

Les polynômes de phase :

$$p_s(x)=\sum_{k=0}^{s} b_k e^{jkx}, \ p_{N-1}=p$$

ont les qualités suivantes :

a. $p(x_k) = f_k = f(x_k)$, $k = 0, 1, \ldots, N-1$, f_k complexe

si *et seulement si :*

$$b_k = \frac{1}{N} \sum_{\ell=0}^{N-1} f_\ell \, e^{-2\pi j \ell k/N}, \; k = 0, 1, \ldots, N-1 \tag{14.1}$$

b. (14.1) a pour conséquence que p_s est la meilleure approximation de f en moyenne quadratique.

p_s est déterminé d'une façon unique par cette propriété.

Nous nous limitons maintenant au cas $N = 2^n$. Cooley et Tukey (1965) ont généralisé une méthode de Gauss pour évaluer rapidement les expressions (14.1) avec $O(N \log N)$ opérations au lieu de $O(N^2)$, nécessité absolue pour effectuer le calcul de la transformée en temps réel.

La transformée discrète est décomposée successivement en deux, quatre, huit, etc. transformées sur $\frac{N}{2}, \frac{N}{4}, \frac{N}{8}$ points jusqu'à deux points (voir par exemple *Microsystèmes* septembre-octobre 1981, pp. 155-159).

Ne pouvant citer tous les détails, nous nous contenterons de dire qu'il faut introduire l'application τ qui inverse la représentation binaire d'un nombre entier k :

$$k = a_0 + a_1 2 + a_2 2^2 + \ldots + a_{n-1} 2^{n-1}, \; a_i = 0 \text{ ou } 1$$

$$\tau(k) = a_{n-1} + a_{n-2} 2 + \ldots + a_0 2^{n-1}$$

on initialise donc par :

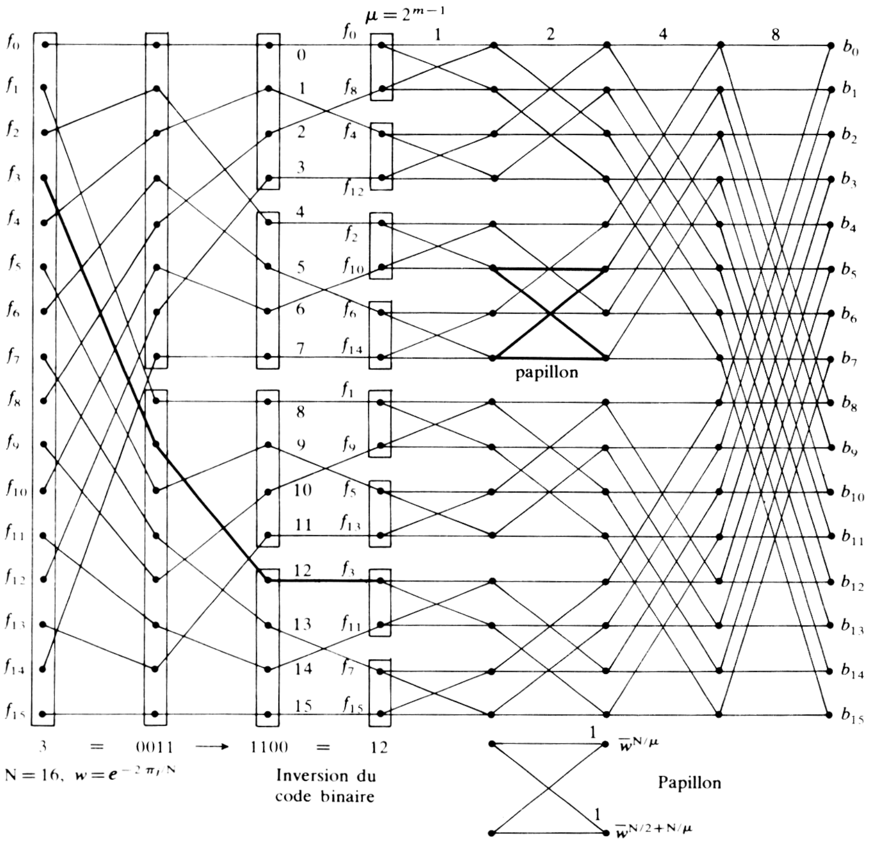
$$b(\tau(k)) = f_k, \; k = 0, \ldots, 2^n - 1$$

et ensuite :

$$m=1, \ldots, n:$$
$$E=1 \quad i=0, \ldots, 2^{m-1}-1$$
$$r=0, 2^m, 2\cdot 2^m, \ldots, (<N-1)$$
$$u:=b(r+i),\ v:=b(r+i+2^{m-1})\cdot E$$
$$b(r+i):=u+v,\ b(r+i+2^{m-1}):=u-v$$

$$E \longleftarrow E \exp\left(\frac{-2\pi j}{2^m}\right)$$

Après exécution, on trouve dans le tableau $b(k)$, $k=0, \ldots, N-1$, les coefficients du polynôme de phase multipliés par N.



ORGANIGRAMME

Entrée des données numériques :

60	N nombre de points d'interpolation $N = 2^{N1}$
110	RB(K) partie réelle de f IB(K) partie imaginaire de f : $RB(K) + j\,IB(K) = f\left(\frac{K*2*PI}{N}\right)$

Inversion binaire :

130 à 185 $\tau(K) = L$, $\tau(L) = K$

Algorithme de Cooley-Tukey :

200 à 320 Calcul des coefficients $b(k)$

Affichage des résultats :

400 à 420 $b(k)$ et $|b(k)|^2$

Dessin du spectre :

430 à 470

PROGRAMME

```
 10 REM *** Transformation de FOURIER -F
FT ***
 20 PRINT"p(x)=b0+b1exp( jx)+b2exp(2jx)+.
..+ b(N-1)exp((N-1)jx) "
 30 PRINT"x(k) = 2PI k/N, (x(k),f(x(k)))
  points d'interpolation, k=0,1,...,N-1"
 40 PRINT"N=2^n, calcul des coefficients
 b(k)"
```

```
 50 REM *** Introduction des valeurs   (
complexes) f(x(k))=f(k) ***
 60 INPUT "Nombre de points d'interpolat
ion N=2^n = ";N
 70 N1 = INT( LOG(N)/LOG(2)+.5): DIMA(N1
)
 80 DIM RB(N-1), IB(N-1)
 90 M1=N/2:L=0
 100 FOR K=0 TO N-1
 105 X=K*2/N
 110 PRINT " Re f(";X;"*PI) , Im f(";X;"
*PI) = ";
 120 INPUT RB(K),IB(K)
 130 IF K<L THEN 170
 140 RB=RB(L):IB=IB(L)
 150 RB(L)=RB(K):IB(L)=IB(K)
 160 RB(K)=RB:IB(K)=IB
 170 J=M1
 175 IF J<=L THEN L=L-J:J=J/2:GOTO 175
 180 L=L+J
 185 NEXT
 190 REM *** Algorithme de COOLEY - TUKE
Y ***
 200 M1=1: FOR M=1 TO N1
 210 A=COS(PI/M1)   : B=-SIN(PI/M1)   :R
E=1: IE=0
 220 FOR J=0 TO M1-1
 230 FOR R = 0 TO N-1 STEP 2*M1
 240 R0=R+J:RM=R0+M1
 250 RU = RB(R0): IU = IB(R0)
 260 RV = RB(RM)*RE - IB(RM)*IE
 270 IV = RB(RM)*IE +IB(RM)*RE
 280 RB(R0) = RU+RV: IB(R0) = IU + IV
 290 RB(RM) = RU-RV :IB(RM) = IU - IV
 300 NEXT
 305 RS=A*RE-B*IE:IE=A*IE+B*RE:RE=RS
 310 NEXT
 315 M1=2*M1
 320 NEXT
 390 PRINT "Coefficients b(k): "
 400 FOR K = 0 TO N-1
```

```
410 PRINT " b(";K;") = ";RB(K)/N;" + j*
"; IB(K)/N;
415 IB(K)=RB(K)*RB(K)/(N*N)+IB(K)*IB(K)
/(N*N): PRINT" |b(";K;")|^2 = ";IB(K)
420 NEXT K
430 PRINT " GRAPHE DU SPECTRE DE PUISSA
NCE DU SIGNAL "
440 FOR K=0 TO N-1
450 PRINTK;SPC(3*INT(N*SQR(IB(K))+.5));
"*"
460 NEXT
470 END
```

DÉROULEMENT DU PROGRAMME

Le programme calcule les coefficients du polynôme de phase $b(k)$. On introduit en ligne 60 le nombre N de points d'interpolation, $N = 2^n$, en ligne 120 les valeurs réelles et imaginaires Ré $f\left(\frac{2\pi k}{N}\right)$, I$m$ $f\left(\frac{2\pi k}{N}\right)$.

On place alternativement ces valeurs dans un tableau où l'on se sert des instructions READ, DATA.

Les coefficients b_k sont liés aux coefficients réels A_k et B_k du polynôme trigonométrique d'interpolation par les relations :

$$A_0 = 2\, b_0,\ A_k = b_k + b_{N-k},\ B_k = j\,(b_k - b_{N-k})$$

Pour calculer la transformée inverse, on emploie le même programme en modifiant :

110 PRINT "Re $b(k)$, Im $b(k)$";

210 $A = COS\left(\frac{PI}{M1}\right)$: $B = SIN\left(\frac{PI}{M1}\right)$

et en supprimant la division par N en ligne 410 :

410 PRINT "$f(2\,pi$"; K; "/"; N;")="; RB(K); "+ $j*$"; IB(K)

EXEMPLES NUMÉRIQUES

Exemple 1

```
p(x)=b0+b1exp(jx)+b2exp(2jx)+...+ b(N-1)
exp((N-1)jx)
x(k) = 2PI k/N, (x(k),f(x(k)))  points d
'interpolation,
k=0,1,...,N-1, N=2^n, calcul des coeffic
ients b(k)
```

```
f(x)=x/pi, 0≤x<2pi
```

```
Re f(0 *PI) , Im f(0 *PI) = 0      0
Re f(.25 *PI) , Im f(.25 *PI) = .25        0
Re f(.5 *PI) , Im f(.5 *PI) = .5  0
Re f(.75 *PI) , Im f(.75 *PI) = .75        0
Re f(1 *PI) , Im f(1 *PI) = 1      0
Re f(1.25 *PI) , Im f(1.25 *PI) = 1.25     0
Re f(1.5 *PI) , Im f(1.5 *PI) = 1.5        0
Re f(1.75 *PI) , Im f(1.75 *PI) = 1.75     0
Coefficients b(k):
b(0 ) = .875  + j*0  |b(0 )|^2 = .766
b(1 ) = -.125  + j*.302  |b(1 )|^2 = .107
b(2 ) = -.125  + j*.125  |b(2 )|^2 = .031
b(3 ) = -.125  + j*.052  |b(3 )|^2 = .018
b(4 ) = -.125  + j*0  |b(4 )|^2 = .016
b(5 ) = -.125  + j*-.052  |b(5 )|^2 = .018
b(6 ) = -.125  + j*-.125  |b(6 )|^2 = .031
b(7 ) = -.125  + j*-.302  |b(7 )|^2 = .107

 GRAPHE DU SPECTRE DE PUISSANCE DU SIGNAL
0                       *
1           *
2     *
3     *
4     *
5     *
6     *
7           *
```

Exemple 2

```
Nombre de Points d'interPolation N=2^n = 8

 f(x)=sin(x)+3sin(3x)-2sin(4x)+.5sin(6x)

 Re f(0 *PI) , Im f(0 *PI) = 0      0
 Re f(.25 *PI) , Im f(.25 *PI) = 2.328     0
 Re f(.5 *PI) , Im f(.5 *PI) = -2  0
 Re f(.75 *PI) , Im f(.75 *PI) = 3.328     0
 Re f(1 *PI) , Im f(1 *PI) = 0      0
 Re f(1.25 *PI) , Im f(1.25 *PI) = -3.328  0
 Re f(1.5 *PI) , Im f(1.5 *PI) = 2         0
 Re f(1.75 *PI) , Im f(1.75 *PI) = -2.328  0
Coefficients b(k):
 b(0 ) = 0  + j*0  |b(0 )|^2 = 0
 b(1 ) = 0  + j*-.5  |b(1 )|^2 = .25
 b(2 ) = 0  + j*.25  |b(2 )|^2 = .062
 b(3 ) = 0  + j*-1.5  |b(3 )|^2 = 2.25
 b(4 ) = 0  + j*0  |b(4 )|^2 = 0
 b(5 ) = 0  + j*1.5  |b(5 )|^2 = 2.25
 b(6 ) = 0  + j*-.25  |b(6 )|^2 = .062
 b(7 ) = 0  + j*.5  |b(7 )|^2 = .25

 GRAPHE DU SPECTRE DE PUISSANCE DU SIGNAL
0 *
1            *
2      *
3                                *
4 *
5                                *
6      *
7            *
```

Exemple 3

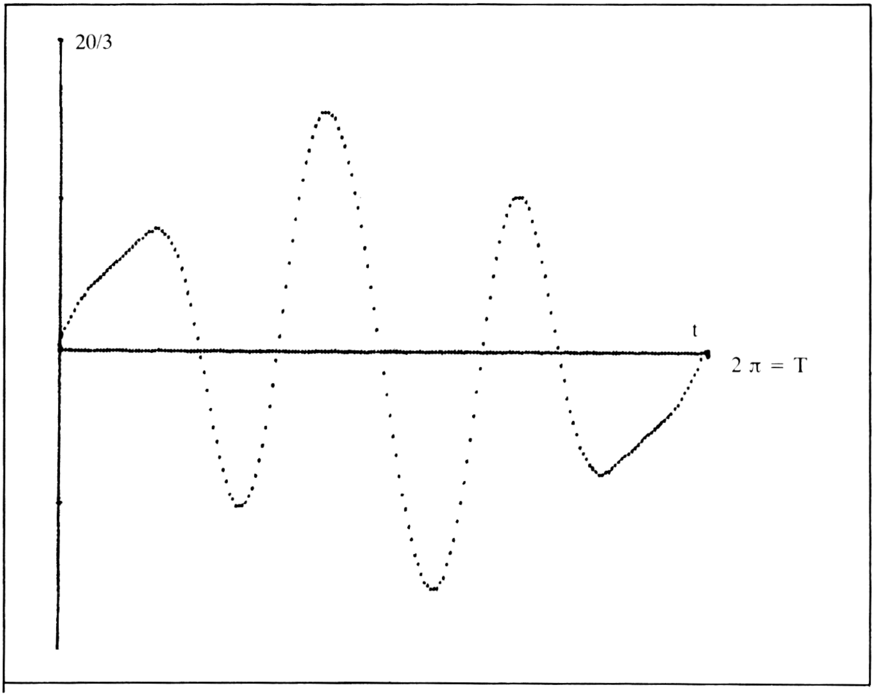
```
p(x)=b0+b1exp(jx)+b2exp(2jx)+...+ b(N-1)
exp((N-1)jx)
x(k) = 2PI k/N, (x(k),f(x(k)))  Points d
'interpolation,
k=0,1,...,N-1, N=2^n, calcul des coeffic
ients b(k)
```

```
Nombre de points d'interpolation N=2^n = 16

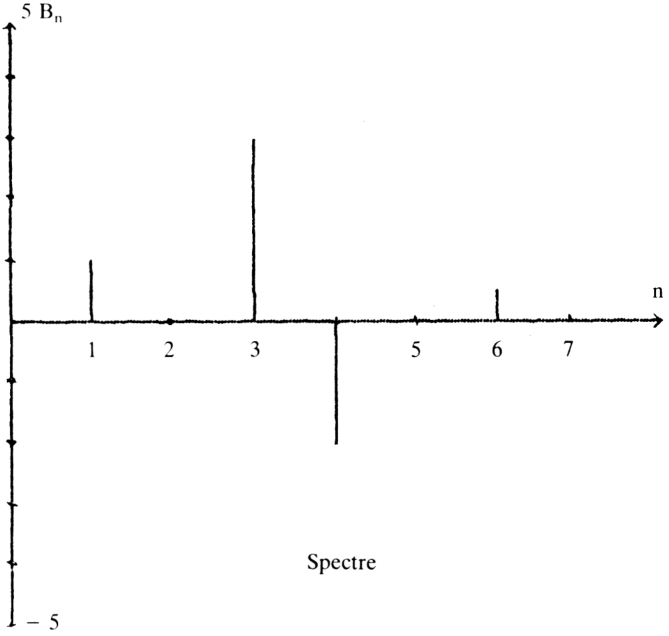
 f(x)=sin(x)+3sin(3x)-2sin(4x)+.5sin(6x)

 Re f(0 *PI) , Im f(0 *PI) = 0      0
 Re f(.125 *PI) , Im f(.125 *PI) = 1.508   0
 Re f(.25 *PI) , Im f(.25 *PI) = 2.328     0
 Re f(.375 *PI) , Im f(.375 *PI) = 2.129   0
 Re f(.5 *PI) , Im f(.5 *PI) = -2  0
 Re f(.625 *PI) , Im f(.625 *PI) = -2.578  0
 Re f(.75 *PI) , Im f(.75 *PI) = 3.328     0
 Re f(.875 *PI) , Im f(.875 *PI) = 4.801   0
 Re f(1 *PI) , Im f(1 *PI) = 0      0
 Re f(1.125 *PI) , Im f(1.125 *PI) = -4.801        0
 Re f(1.25 *PI) , Im f(1.25 *PI) = -3.328  0
 Re f(1.375 *PI) , Im f(1.375 *PI) = 2.578         0
 Re f(1.5 *PI) , Im f(1.5 *PI) = 2         0
 Re f(1.625 *PI) , Im f(1.625 *PI) = -2.129        0
 Re f(1.75 *PI) , Im f(1.75 *PI) = -2.328  0
 Re f(1.875 *PI) , Im f(1.875 *PI) = -1.508        0
Coefficients b(k):
 b(0 ) = 0  + j*0  |b(0 )|^2 = 0
 b(1 ) = 0  + j*-.5  |b(1 )|^2 = .25
 b(2 ) = 0  + j*0  |b(2 )|^2 = 0
 b(3 ) = 0  + j*-1.5  |b(3 )|^2 = 2.25
 b(4 ) = 0  + j*1  |b(4 )|^2 = 1
 b(5 ) = 0  + j*0  |b(5 )|^2 = 0
 b(6 ) = 0  + j*-.25  |b(6 )|^2 = .062
 b(7 ) = 0  + j*0  |b(7 )|^2 = 0
 b(8 ) = 0  + j*0  |b(8 )|^2 = 0
 b(9 ) = 0  + j*0  |b(9 )|^2 = 0
 b(10 ) = 0  + j*.25  |b(10 )|^2 = .062
 b(11 ) = 0  + j*0  |b(11 )|^2 = 0
 b(12 ) = 0  + j*-1  |b(12 )|^2 = 1
 b(13 ) = 0  + j*1.5  |b(13 )|^2 = 2.25
 b(14 ) = 0  + j*0  |b(14 )|^2 = 0
 b(15 ) = 0  + j*.5  |b(15 )|^2 = .25
```

```
    GRAPHE DU SPECTRE DE PUISSANCE DU SIGNAL
0  *
1                         *
2  *
3                                                                       *
4                                                **
5  *
6             *
7  *
8  *
9  *
10 *
11 *
12                                               *
13                                                                      *
14 *
15                        *
```



g(t)=sin(t)+3sin(3t)-2sin(4t)+.5sin(6t)

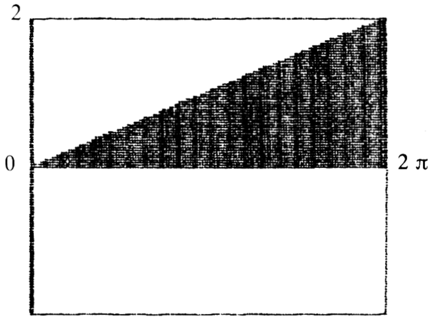


Figures 14-1

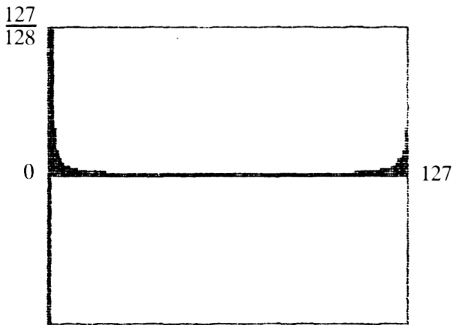
```
P(x)=b0+b1exP(jx)+b2exP(2jx)+...+ b(N-1)exP((N-1)jx)
x(k) = 2PI k/N, (x(k),f(x(k))) Points d'interPolation,
k=0,1,...,N-1, N=2^n, calcul des coefficients b(k)
Nombre de Points d'interPolation N=2^n = 128

max |b(k)| = 127/N
```

f (x) = x/π



Spectre | b (k) |



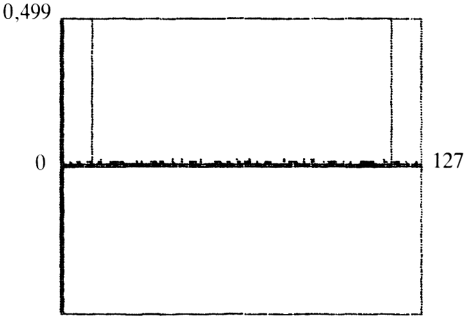

```
f(x) = sin(10*x) + bruit

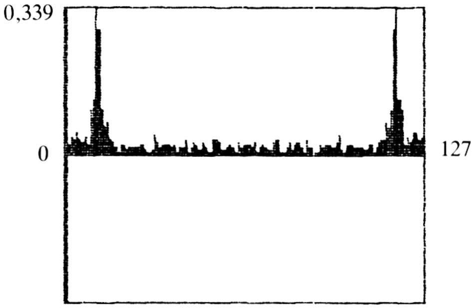
max |b(k)| = 63.9284486/N
```

```
f(x) = sin(10.5*x) + bruit

max |b(k)| = 43.3346085/N
```

Spectres $|b(k)|$:





Figures 14.2

V
ÉQUATIONS DIFFÉRENTIELLES

15. ÉQUATIONS DIFFÉRENTIELLES DU SECOND ORDRE NON LINÉAIRES

INITIATION AU PROBLÈME

Il existe en électronique une classe de montages appelés oscillateurs et qui jouent un rôle important. Ces montages ont pour but de fournir sur une de leurs bornes un signal sinusoïdal dont on maîtrise en général la fréquence.

Il s'agit de systèmes bouclés, dont la réaction est positive, et qui obéissent au critère de Barkhausen : le gain de boucle doit être égal à $+1$ pour une certaine fréquence, celle de l'oscillation justement. Or, il est difficile de réaliser une telle condition de façon parfaite. On s'arrange d'ailleurs en général pour que ce gain de boucle soit légèrement supérieur à $+1$ et que l'on soit sûr, par conséquent, qu'il y a oscillation.

Par ailleurs, l'amplificateur (nécessaire) n'a pas une fonction de transfert parfaite du type $v_1 = K v_2$, mais il présente des paliers de saturation : $\pm$Vsat.

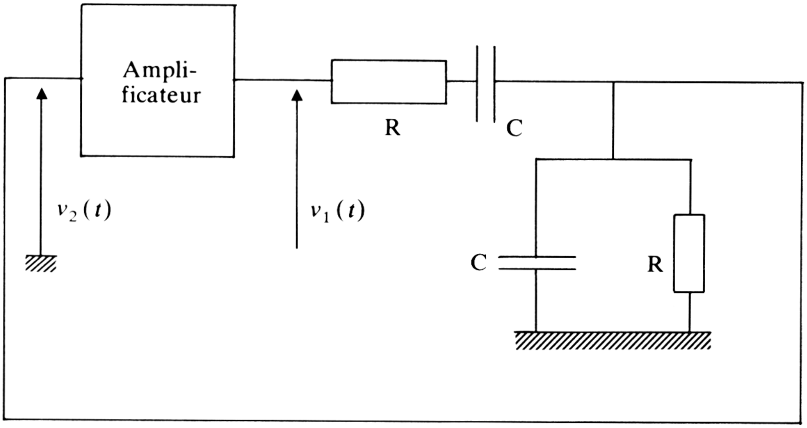


Figure 15.1.

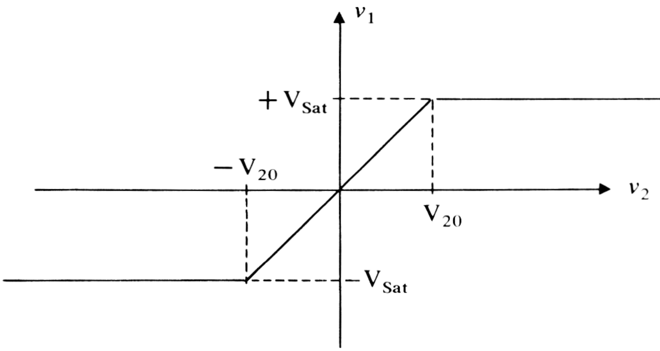


Figure 15.2 : Fonction de transfert d'un amplificateur réel.

La partie linéaire de cette fonction correspond au *gain* de l'amplificateur. Il est possible de rendre compte de l'allure de cette caractéristique en posant :

$$v_1 = V_{sat} \tanh \left(\frac{v_2}{V_{20}}\right)$$

ce qui est d'ailleurs proche de la réalité, le dessin ci-dessus n'étant en fait qu'une modélisation de l'amplificateur.

Cette relation, jointe à l'équation différentielle du Pont de Wien (voir encadré), fournit l'équation régissant le fonctionnement d'un oscillateur à Pont de Wien.

L'équation devient alors ($v_2 =: v$) :

$$\frac{d^2 v}{dt^2} + \frac{3}{\tau}\frac{dv}{dt} - \frac{V_{sat}}{\tau V_{20}} \frac{1}{\cosh^2 \dfrac{v}{V_{20}}} \cdot \frac{dv}{dt} + \frac{1}{\tau^2}\, v = 0$$

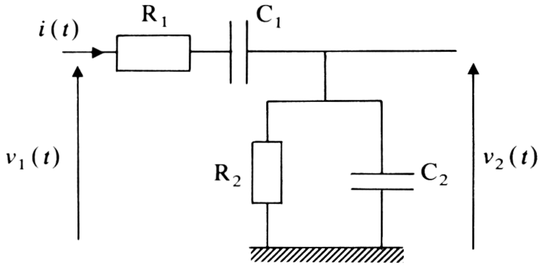
et on a une équation différentielle de Liénard si :

$$\frac{V_{sat}}{V_{20}} = 3 + \varepsilon,\ \varepsilon > 0$$

Cette équation possède une solution périodique stable v_p (sans prendre en considération les translations de temps $t \longrightarrow t + Dt$) dont toutes les autres solutions s'approchent (voir la Figure 15.3).

Nous allons donc étudier le traitement numérique de cette équation.

Équation différentielle du Pont de Wien



$$i = C_2 \frac{dv_2}{dt} + \frac{v_2}{R_2} \qquad \frac{di}{dt} = C_2 \frac{d^2 v_2}{dt^2} + \frac{1}{R_2} \frac{dv_2}{dt}$$

et :

$$i = C_1 \frac{d}{dt}(v_1 - R_1 i - v_2)$$

D'où :

$$C_2 \frac{dv_2}{dt} + \frac{v_2}{R_2} = C_1 \frac{dv_1}{dt} - R_1 C_1 \left(C_2 \frac{d^2 v_2}{dt^2} + \frac{1}{R_2} \cdot \frac{dv_2}{dt} \right) - C_1 \frac{dv_2}{dt}$$

et :

$$\frac{d^2 v_2}{dt^2} + \frac{dv_2}{dt} \left(\frac{1}{R_1 C_1} + \frac{1}{R_2 C_2} + \frac{1}{R_1 C_2} \right) - \frac{1}{R_1 C_2} \frac{dv_1}{dt} + \frac{1}{R_1 R_2 C_1 C_2} v_2 = 0$$

On pose en général, pour simplifier :

$$\left. \begin{array}{l} R_1 = R_2 = R \\ C_1 = C_2 = C \end{array} \right\} RC = \tau$$

D'où :

$$\frac{d^2 v_2}{dt^2} + \frac{3}{\tau} \frac{dv_2}{dt} - \frac{1}{\tau} \frac{dv_1}{dt} + \frac{1}{\tau^2} v_2 = 0$$

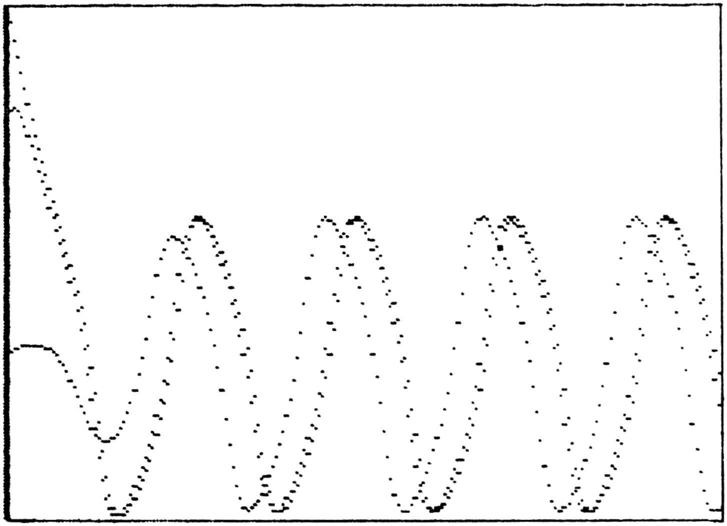


Figure 15.3.

Évoquons encore un autre aspect :

Une des équations fondamentales de la physique est la célèbre relation de Newton :

$$F = ma$$

$\Bigl($ m masse d'un corps; F force qui peut résulter d'un potentiel lié à la masse; a accélération, $a = \dfrac{d^2 y}{dx^2} \cdot \Bigr)$

$y(x)$ donne la position du corps, $y(x_0)$ sa position à l'instant x_0, $y'(x_0)$ sa vitesse initiale, x le temps.

Comme F est une fonction des variables x, y, y', notre problème s'exprime au moyen d'une équation différentielle du second ordre :

$$y'' = \frac{F}{m} = f(x, y, y')$$

et deux conditions initiales :

$$y(x_0) = y_0, \quad y'(x_0) = u_0$$

ou si l'on pose :

$y = y_1, \quad y' = y_2$, par un système

$$y_1'(x) = y_2(x), \qquad y_1(x_0) = y_{01}$$

$$y_2'(x) = f(x, y_1(x), y_2(x)), \quad y_2(x_0) = y_{02}$$

MÉTHODE

Tracé d'une solution de l'équation $y'' = f(x, y, y'')$.

Nous discutons d'abord le problème à valeurs initiales du second ordre :

$$y'' = f(x, y, y'),\ y(a) = y_0,\ y'(a) = u_0$$

et nous voulons tracer sur l'écran la courbe solution $y(x)$ ou sa dérivée $y'(x)$, $a \leqq x \leqq b$, et afficher les valeurs approchées dans un tableau.

Dans ce but, nous partageons le segment $[a, b]$ en 200 intervalles partiels égaux et nous utilisons sur chacun d'eux une formule de Runge-Kutta d'ordre 5 (erreur locale $= O(n^5)$).

Partons donc du point initial x_0, y_0, u_0 et posons $h = \dfrac{(b-a)}{200}$:

$$L1 = hf(x_0, y_0, u_0)$$

$$x = x_0 + \frac{h}{2},\ y = y_0 + \frac{hu_0}{2},\ y' = u_0 + \frac{L1}{2}$$

$$L2 = hf(x, y, y')$$

$$x = x_0 + \frac{h}{2},\ y = y_0 + \frac{hu_0}{2} + \frac{L1\,h}{4},\ y' = u_0 + \frac{L2}{2}$$

$$L3 = hf(x, y, y')$$

$$x = x_0 + h, \ y = y_0 + hu_0 + \frac{L2\,h}{2}, \ y' = u_0 + L3$$

$$L4 = hf(x, y, y')$$

et finalement le procédé repart à partir de :

$$x_0 \longleftarrow x_0 + h, \ y_0 \longleftarrow y_0 + hu_0 + \frac{h(L1 + L2 + L3)}{6}$$

$$u_0 \longleftarrow u_0 + \frac{(L1 + 2L2 + 2L3 + L4)}{6}$$

Si l'intervalle $[a, b]$ est important, un partage est recommandé.

Système d'équations différentielles d'ordre 1 d'après Fehlberg (*)

Nous allons étudier le système d'équations différentielles :

$$y'_k = f_k(x, y_1, y_2, \ldots, y_n), \ k = 1, 2, \ldots, n, \ a \leqq x \leqq b$$

ou en forme vectorielle :

$$u = y' = f(x, y)$$

Soit x_0, y_0 le point initial (x_0, I) de la courbe solution. L'équation différentielle nous donne alors la pente $f(x_0, y_0)$ de la tangente à la solution.

---

* E. Fehlberg : *Klassische Runge-Kutta Formeln vierter und niedrigerer Ordnung mit Schrittweitenkontrolle und ihre Anwendung auf Wärmeleitungsprobleme.* Computing 6 (1970), pp. 61-71.

J. Legras : *Méthodes et Techniques de l'analyse numérique*, Dunod, Paris, 1971.

L'approximation consiste à suivre la tangente au lieu de la courbe sur un petit intervalle $[x_0, x_0 + h]$, h pas d'intégration.

Nous posons :

$$\mathrm{L} = hf(x_0, y_0)$$

Pour améliorer l'approximation, nous reculons et nous recommençons le même procédé à partir du point :

$$x_0 + a_1 h, \ y_0 + b_1 \mathrm{L}$$

Cela nous donne :

$$\mathrm{M} = hf(x_0 + a_1 h, \ y_0 + b_1 \mathrm{L})$$

et le prochain point de départ sera :

$$x_0 + a_2 h, \ y_0 + b_2 \mathrm{L} + b_3 \mathrm{M}$$

etc. La valeur approchée $y_0 + \mathrm{D}y$ de la courbe solution devient finalement :

$$y_0 + \mathrm{D}y = y_0 + c_1 \mathrm{L} + c_2 \mathrm{M} + c_3 \mathrm{Q}$$

Nous choisissons donc une formule à trois (quatre) approximations (voir J. Legras, p. 177) et nous déterminons les constantes a_i, b_i, c_i de telle manière que :

- l'écart entre la valeur exacte y et la valeur approchée soit infiniment petit d'ordre 3 (4);
- les trois premières approximations des deux formules d'ordre 3 et 4 soient identiques.

La quatrième O servira de première approximation L du prochain pas : $x_0 + h \longrightarrow x_0 + 2h$

Fehlberg propose le schéma suivant :

	a_i	b_i			c_i	$\tilde{c}_i$
L	0	0			$\frac{214}{891}$	$\frac{533}{2\,106}$
M	$\frac{1}{4}$	$\frac{1}{4}$			$\frac{1}{33}$	0
Q	$\frac{27}{40}$	$\frac{-189}{800}$	$\frac{729}{800}$		$\frac{650}{891}$	$\frac{800}{1\,053}$
O	1	$\frac{214}{891}$	$\frac{1}{33}$	$\frac{650}{891}$	S	$\frac{-1}{78}$

$$E = -\frac{23\,L}{1\,782} + \frac{M}{33} - \frac{350\,Q}{11\,583} + \frac{O}{78}$$

Celui-ci ne diffère guère d'un schéma proposé par Mineur et donne fort avantageusement l'erreur systématique E par pas. Cela nous permet de contrôler et d'ajuster la valeur de h au cours du calcul sans perte de temps. Si E est trop grand, $|E| > P$, P erreur locale, on rejette la valeur approchée $y_0 + Dy$ et on remplace h par $\frac{h}{2}$, puis on recommence le calcul à partir de y_0. Si E est de très faible valeur, $|E| < \frac{P}{100}$, on remplace h par $2\,h$.

PROGRAMMES

```
 10 REM *** Resolution d'un systeme d' e
quations differentielles d'ordre un ***
 20 INPUT " Nombre d'equations n = ";N
 30 INPUT " Intervalle [a,b] = ";A,B
 40 IF A>=B THEN PRINT "a>=b??": GOTO 30
 50 DIM Y(N),U(N),S(N),I(N),E(N),L(N), M
(N),O(N),Q(N)
```

```
 60 FOR K=1 TO N
 70 READ I(K): REM *** Valeurs initiales
 ***
 80 Y(K)=I(K):S(K)=I(K)
 90 NEXT
 100 INPUT " Pas d'integration h = ";H0
 110 IF H0<=0 THEN PING: GOTO 100
 120 INPUT " Erreur locale P= "; P
 130 IF P <= 0 THEN PING: GOTO 120
 140 X0 = A:H=H0:X=X0:G=0
 150 GOSUB 700
 160 FOR K=1TO N
 170 L(K)=H*U(K):E(K)=0
 180 NEXT
 190 PRINT" Valeurs x, y(1),...,y(n) :"
 200 PRINT X0;
 210 FOR K=1TO N
 220 PRINT " ";I(K);
 230 NEXT
 240 PRINT
 250 IF X0>=B   THEN END
 260 FOR K=1 TO N
 270 S(K) = S(K) +214*L(K)/891:E(K)=E(K)
-23*L(K)/1782:Y(K)=I(K)+L(K)/4
 280 NEXT
 290 X=X+H/4
 300 GOSUB 700
 310 FOR K=1 TO N
 320 M(K)=H*U(K):S(K)=S(K)+M(K)/33:E(K)=
E(K)+M(K)/33
 325 Y(K)=I(K)-189*L(K)/800+729*M(K)/800
 330 NEXT
 340 X=X+H*17/40
 350 GOSUB 700
 360 FOR K=1 TO N
 370 Q(K)=H*U(K):S(K)=S(K)+650*Q(K)/891:
E(K)=E(K)-350*Q(K)/11583:Y(K)=S(K)
 380 NEXT
 390 X=X+H*13/40
 400 GOSUB 700
 410 FOR K = 1 TO N
 420 O(K) = H*U(K): E(K)=E(K)+O(K)/78
```

```
 430 NEXT
 440 REM *** Precision obtenue ? ***
 450 K=0
 460 K=K+1: IF K>N THEN 500
 470 IF ABS(E(K))>P THEN 600
 480 IF ABS(E(K))<P/100 THEN 460
 490 G=1:GOTO 460
 500 IF G=0 AND H<H0 THEN H=2*H:GOTO 510
 505 IF H>=H0 THEN G=1
 510 FOR K=1 TO N
 520 I(K) = S(K):L(K)=(2-G)*O(K):E(K)=0
 530 NEXT
 540 G=0: X0=X
 550 GOTO 200
 600 H=H/2: X=X0: G=0
 610 FOR K=1 TO N
 620 E(K)=0:S(K)=I(K):L(K)=L(K)/2
 630 NEXT
 640 GOTO 260
 690 REM *** Systeme d'equations ***
 700 U(1) = (Y(1)+Y(2))/(2*X)
 710 U(2) = (Y(1)-Y(2))/(2*X)
 720 RETURN
 800 REM *** Valeurs initiales y0(1),...
y0(n) ***
 810 DATA 1,0
```

```
 10 REM *** Trace d'une solution ***
 12 REM *** de l'equation differen-***
 14 REM *** tielle y''=f(x,y,y')***
 16 REM *** y(a)=y0, y'(a)=u0, a<x<b***
 18 PRINT"y'' = -sin(y)"
 19 LPRINT "y'' = -sin(y)"
 20 PRINT"Intervalle [a,b] = ";:INPUT A,
B : PRINT "[";A;",";B;"]"
 25 LPRINT "Intervalle [a,b] = ";"[";A;"
,";B;"]"
 30 IF A>=B THEN PING: GOTO 20
 40 PRINT "Valeur initiale y(a) = ";:INP
UT Y0: PRINT Y0
```

```
 45 LPRINT "Valeur initiale y(a) = "; Y0
 50 PRINT "Valeur initiale y'(a)= ";:INP
UT U0: PRINT U0
 51 LPRINT "Valeur initiale y'(a)= "; U0
 55 LPRINT:PRINT
 60 INPUT "Trace y(x) ou y'(x)? 1/2 ";A$
: T=VAL(A$)
 70 IF A$<>"1" THEN IF A$<>"2" THEN PING
: GOTO 60
 75 IF C$="O" THEN 85
 80 DIMV(200)
 85 M1=Y0: M2=Y0:V(0)=Y0:IF VAL(A$)=2 TH
EN M1=U0:M2=U0:V(0)=U0
 90 REM *** Pas d'integration h=(b-a)/20
0***
 100 REM *** x(k)= a+k*(b-a)/200, V(k)=y
(x(k)) ou y'(x(k)) ***
 110 REM *** V(K) Fichier ***
 120 H=(B-A)/200: X0=A:K=0
 130 REM *** Methode de RUNGE-KUTTA ***
 140 X=X0: Y=Y0:U =U0
 150 GOSUB 500
 160 L1=H*F
 170 X=X0+H/2:Y=Y0+H*U0/2:U =U0+L1/2
 180 GOSUB 500
 190 L2=H*F
 200 Y=Y0+H*U0/2+L1*H/4: U =U0+L2/2
 210 GOSUB 500
 220 L3=H*F
 230 X=X0+H: Y=Y0+H*U0+H*L2/2:U =U0+L3
 240 GOSUB 500
 250 L4=H*F
 260 X0=X0+H:Y0=Y0+H*U0+H*(L1+L2+L3)/6
 270 U0=U0+(L1+2*L2+2*L3+L4)/6
 280 K=K+1
 290 IF A$="2" THEN 330
 300 V(K)=Y0:IF Y0<M1 THEN M1=Y0
 310 IF Y0>M2 THEN M2=Y0
 320 GOTO 350
 330 V(K)=U0: IF U0<M1 THEN M1=U0
 340 IF U0>M2 THEN M2=U0
```

```
 350 IF K<200 THEN 140
 355 IF C$="O" THEN 370
 360 HIRES: REM *** Trace ***
 365 M3=M1:M4=M2
 370 FOR K=0 TO 200
 380 V%=160*(M4-V(K))/(M4-M3)
 390 CURSET K,V%,1
 400 NEXT
 410 REM *** Axes ***:IF C$="O" THEN 475
 420 V% = 160*(M4-V(0))/(M4-M3)
 430 CURSET 0,V%,1: DRAW 200,0,1
 440 CURSET 0,0,1 : DRAW 0,160,1
 445 CURSET 10,170,0
 450 N$ = "Courbe y(x) - trace ":IF T=
2  THEN N$ = "Courbe y'(x) - trace "
 455 FOR K=1 TO LEN(N$)
 460 CHAR ASC(MID$(N$,K,1)),0,1
 465 CURMOV 10,0,0
 470 NEXT
 474 REM POKE26,96 permet d'ecrire dans
les 3 lignes en bas de l'ecran Haute Res
 475 POKE 26,96:PRINT "Valeurs extremes
:";M1;" ";M2;" ";
 476 LPRINT "Valeurs extremes :";M1;" ";
M2;" ":LPRINT
 480 PRINT"Affichage du tableau des vale
urs O/N?";:GET A$
 490 IF A$= "N" THEN GOTO 640 ELSE GOTO
600
 495 REM *** Equation differentielle
            F = F(X,Y,U) ***
 500 F=-SIN(Y)
 510 RETURN
 600 LPRINT " Tableau des valeurs x(k),
y(k)(y'(k):"
 610 FOR K=0 TO 200
 620 LPRINT K,A+K*(B-A)/200, V(K)
 630 NEXT
 635 GOTO 660
 640 PRINT" Autres valeurs initiales? (O
/N)":GET C$
 650 IF C$="O" THEN 40
```

```
 660 PRINT" Recopie ecran sur imprimante
? (O/N)": GET D$
 670 IF D$<>"N" THEN GOTO 1000
 680 END
 990 REM *** Imprimante ***
 1000 PRINT"AJOUTER SOUSPROGRAMME RECOPI
E ECRAN HAUTE RESOLUTION"
 1005 WAIT 500
 1010 END
```

UTILISATION DES PROGRAMMES

Tracé d'une solution de l'équation $y'' = f(x, y, y')$

On introduit la fonction en lignes 500 et 19 :

500 $\quad F = f(X, Y, U)$

$F \mathrel{\widehat{=}} y''$, $X \mathrel{\widehat{=}} x$, $Y \mathrel{\widehat{=}} y$, $U \mathrel{\widehat{=}} y'$

Le programme demande (lignes 20, 40, 50, 60) :

- l'intervalle $[a, b]$ (INPUT A, B);
- les valeurs initiales $y(a)$ et $y'(a)$ (INPUT Y0, INPUT U0);
- de tracer y ou y' (1/2).

Le pas d'intégration est fixe : $h = \dfrac{(b-a)}{200}$.

Un fichier de 201 valeurs approchées de la fonction y ou y' est dressé et les valeurs extrêmes M1 et M2 sont retenues. On peut choisir entre :

- l'affichage du tableau des valeurs approchées (ligne 480);
- le tracé sur écran (les axes passent par le point initial).

Il est possible de superposer plusieurs courbes si les valeurs extrêmes des courbes suivantes ne dépassent pas celles de la première à tracer.

Une mise en page sur l'imprimante est proposée.

Résolution d'un système

Il faut introduire en lignes :

20 le nombre d'équations N, $1 \leqq N$

30 l'intervalle $[a, b]$: A, B, A<B

100 le pas d'intégration h : H0 $0<h<0,2$

120 l'erreur locale p : P $0<p<10^{-3}$

les N fonctions du système

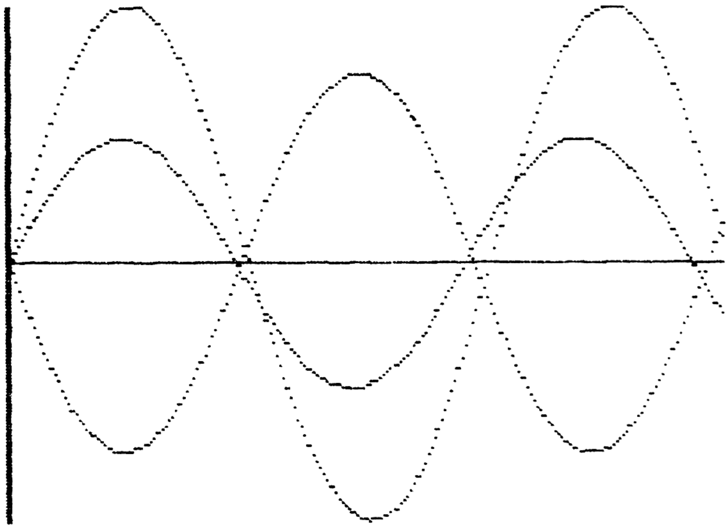
700 $U1 = f_1(X, Y(1), ..., Y(N))$: ... : $U(N) = f_n(X, Y(1), ..., Y(N))$

les N valeurs initiales $y0(1), ..., y0(N)$

810 DATA.,.,...,.

EXEMPLES NUMÉRIQUES

Pendule pesant



Courbe y(x) - trace

```
y'' = -sin(y)
Intervalle [a,b] = [0 ,10 ]
Valeur initiale y(a) = 0
Valeur initiale y'(a)= 1

Valeurs extremes :-1.04717477  1.04710962

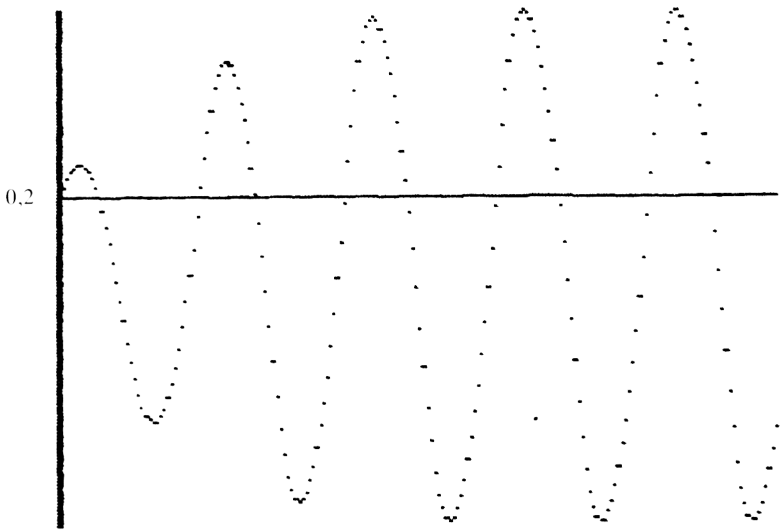
Valeur initiale y(a)= 0
Valeur initiale y'(a)= .5

Valeurs extremes :-.505329743  .50535709

Valeur initiale y(a) = 0
Valeur initiale y'(a)= -.75

Valeurs extremes :-.768787011  .768774499
```

Pont de Wien



Courbe y(x) - trace

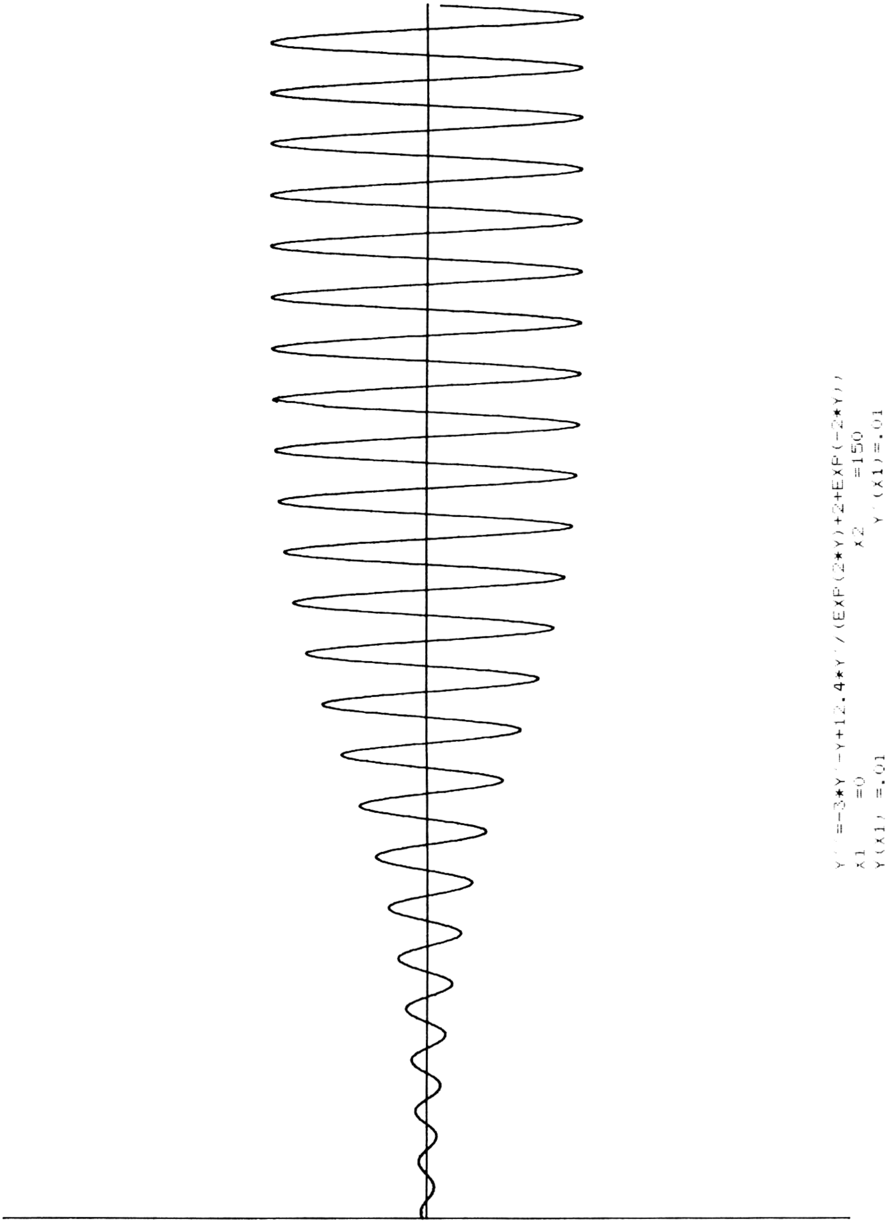
```
y''+3y'(1-1.133/cosh^2(y))+y=0
Intervalle [a,b] = [0 ,30 ]
Valeur initiale y(a) = .2
Valeur initiale y'(a)= .2

Valeurs extremes :-.746286505  .746483
```

```
y'(1)=(y(1)+y(2))/(2x)
y'(2)=(y(1)-y(2))/(2x)
```

```
Nombre d'equations n = ? 2
Intervalle [a,b] = ? 1,2
Pas d'integration h = ? .05
Erreur locale p= ? 1E-07
Valeurs x, y(1),...,y(n) :
1   1   0
1.05   1.02499504   .0244001748
1.1   1.04996294   .0476916235
1.15   1.07488197   .0699953831
1.2   1.09973517   .0914142399
1.25   1.12450926   .112036135
1.3   1.14919393   .131936823
1.35   1.17378116   .151181975
1.4   1.19826484   .169828846
1.45   1.22264031   .18792763
1.5   1.24690413   .205522543
1.55   1.27105384   .222652721
1.6   1.29508776   .239352953
1.65   1.31900486   .255654286
1.7   1.34280461   .271584539
1.75   1.36648692   .287168722
1.8   1.39005205   .302429397
1.85   1.41350052   .317386983
1.90000001   1.43683308   .332060013
1.95000001   1.46005068   .346465359
2.00000001   1.48315439   .360618419
```

```
Nombre d'equations n = ? 2
Intervalle [a,b] = ? 1,1.2
Pas d'integration h = ? .2
Erreur locale p= ? 1E-07
Valeurs x, y(1),...,y(n) :
1   1   0
1.05   1.02499504   .0244001748
1.1   1.04996294   .0476916235
1.15   1.07488197   .0699953831
1.2   1.09973517   .0914142399
```


Y''=-3*Y'-Y+12.4*Y'/(EXP(2*Y)+2+EXP(-2*Y))
X1 =0
X2 =150
Y(X1)=.01
Y'(X1)=.01

16. ÉQUATIONS DIFFÉRENTIELLES AUX DÉRIVÉES PARTIELLES

ÉQUATION PARABOLIQUE

C'est une équation du type conduction de la chaleur :

$$\frac{\partial \theta}{\partial \mathrm{T}} = \alpha \frac{\partial^2 \theta}{\partial \mathrm{X}^2}$$

qui donne la température θ en un point d'abscisse X, au temps T, d'une fine barre métallique qui ne transmet la chaleur que par ses extrémités.

Il est intéressant dans un premier temps de normaliser cette équation de façon que toutes celles qui ont la même forme se résolvent par la même méthode. Posons :

$$\varphi = \frac{\theta}{\theta_0} \quad \text{et} \quad u = \frac{\mathrm{X}}{\mathrm{X}_0}$$

X_0 et θ_0 étant des constantes pouvant avoir une signification physique. Alors :

$$\frac{\partial \theta}{\partial \mathrm{X}} = \frac{\partial \theta}{\partial u} \times \frac{\partial u}{\partial \mathrm{X}} = \frac{\partial \theta}{\partial u} \times \frac{1}{\mathrm{X}_0}$$

$$\frac{\partial^2 \theta}{\partial \mathrm{X}^2} = \frac{\partial}{\partial \mathrm{X}} \left(\frac{\partial \theta}{\partial \mathrm{X}} \right) = \frac{\partial}{\partial u} \left(\frac{1}{\mathrm{X}_0} \cdot \frac{\partial \theta}{\partial u} \right) \cdot \frac{\partial u}{\partial \mathrm{X}} = \frac{1}{\mathrm{X}_0^2} \cdot \frac{\partial^2 \theta}{\partial u^2}$$

D'où, en reprenant l'équation de départ :

$$\theta_0 \frac{\partial \varphi}{\partial \mathrm{T}} = \alpha \times \frac{1}{\mathrm{X}_0^2} \times \frac{\partial^2 \varphi}{\partial u^2} \times \theta_0$$

Enfin, posant $t = \frac{\alpha T}{X_0^2}$, on aboutit à :

$$\frac{\partial \varphi}{\partial t} = \frac{\partial^2 \varphi}{\partial u^2}$$

Nous allons maintenant évaluer numériquement ces dérivées partielles.

Par définition d'une dérivée :

$$\frac{\partial \varphi}{\partial t} \simeq \frac{\varphi(t+k) - \varphi(t)}{k}$$

en appelant k un accroissement de t *assez petit* (nous précisons plus loin cette notion).

De même, nous pouvons écrire :

$$\frac{\partial \varphi}{\partial u} \simeq \frac{\varphi(u+h) - \varphi(u)}{h}$$

et tout aussi bien :

$$\frac{\partial \varphi}{\partial u} \simeq \frac{\varphi(u) - \varphi(u-h)}{h}$$

Or, par définition :

$$\frac{\partial^2 \varphi}{\partial u^2} \simeq \frac{\frac{\partial \varphi}{\partial u}(u) - \frac{\partial \varphi}{\partial u}(u-h)}{h}$$

D'où :

$$\frac{\partial^2 \varphi}{\partial u^2} \simeq \frac{\varphi(u+h) - 2\,\varphi(u) + \varphi(u-h)}{h^2}$$

L'équation de départ nous conduit ainsi à la relation :

$$\frac{\varphi(u,\ t+k) - \varphi(u,\ t)}{k} \cong \frac{\varphi(u+h,\ t) - 2\,\varphi(u,\ t) + \varphi(u-h,\ t)}{h^2}$$

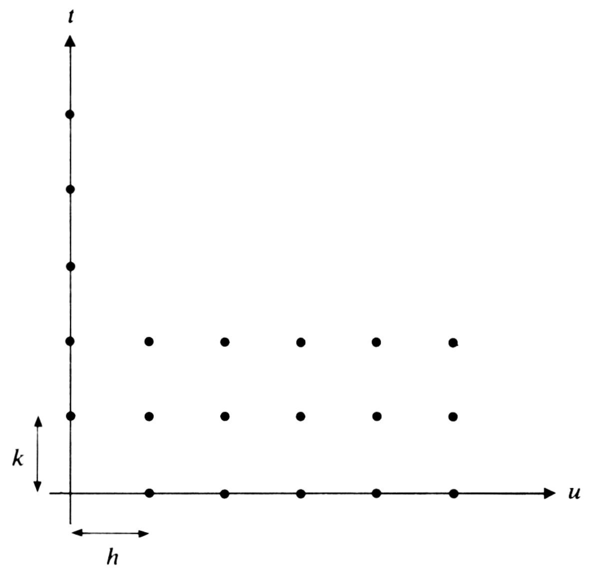


Figure 16.1.

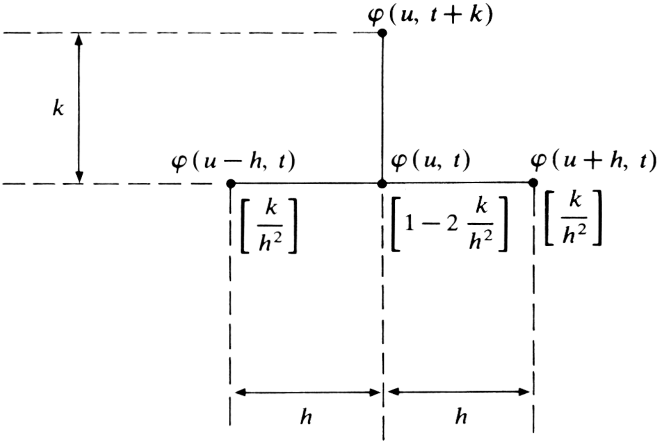


Figure 16.2.

Considérons maintenant une *maille* repérée en abscisse par u et en ordonnée par t. En chaque point de cette maille, de coordonnées $u = ih$ (i : entier positif ou nul), $t = jk$ (k entier positif ou nul), existe une valeur de $\varphi = \varphi(u, t)$.

Or, l'équation précédente nous permet d'évaluer $\varphi(t+k)$ au point d'abscisse u, connaissant $\varphi(t)$ en u et $\varphi(u+h)$, $\varphi(u)$, $\varphi(u-h)$ à l'instant t.

On peut ainsi écrire, de façon plus explicite :

$$\varphi(u, t+k)=\varphi(u, t)+\frac{k}{h^2}(\varphi(u+h, t)-2\varphi(u, t)+\varphi(u-h, t))$$

(Figure 16.2).

L'évolution du système peut être ainsi maîtrisée instant après instant, connaissant :

- les conditions initiales : répartition des températures le long de la barre à l'instant initial $t=0$;
- les effets de « bord » : températures imposées à chaque instant aux deux extrémités de la barre.

Tout le problème est maintenant de réaliser un choix judicieux de h et de k. Les variables du système étant réduites, il paraît raisonnable de choisir h et k petits devant l'unité. Il a alors été démontré que la méthode n'est convergente que si :

$$\frac{k}{h^2}\leqslant 0{,}5$$

ce qui est fort gênant, parce qu'un h de l'ordre de 10^{-2} conduit à un k 200 fois plus petit.

Application

La loi étudiée, de la forme :

$$\frac{\partial \varphi}{\partial t}=\frac{\partial^2 \varphi}{\partial u^2}$$

est très généralement connue par les physiciens sous le nom de loi de

Diffusion (ou seconde loi de Fick); elle exprime la conservation du nombre de particules; elle s'écrit :

$$\frac{\partial n}{\partial t} = D \frac{\partial^2 n}{\partial u^2}$$

dans le cas unidimensionnel :

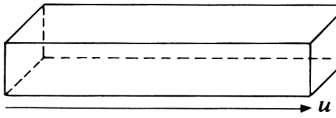


Figure 16.3.

Remarque

Une solution avec $n(o, t) = 0$ et $n(u, 0) = 1$ est $n(u, t) : =$

$$\Phi\left(\sqrt{\frac{D}{2t}}\, u\right)$$

$\Phi =$ fonction d'erreur (voir le Chapitre 5).

Soit n la concentration des molécules d'un gaz dans un conduit parallélépipédique. Si cette concentration n'est pas homogène à $t = 0$, elle est appelée à varier en fonction du temps; cette évolution est régie par la loi précédente.

Considérons l'exemple suivant :

Soit un conduit rectiligne reliant deux récipients entre eux :

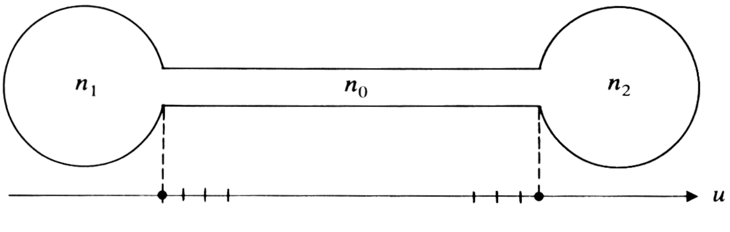


Figure 16.4.

A $t = 0$, la densité de molécules est homogène dans le tube et vaut n_0. Elle est imposée égale à n_1 (à gauche) et à n_2 (à droite), quel que soit le temps t.

Le problème est de savoir comment va varier, à partir de $t=0$, la densité $n(u, t)$ en chaque point du conduit. Le régime permanent peut être connu par un raisonnement simple :

Si, au bout d'un temps infini, s'établit un régime stationnaire, c'est que l'on a alors :

$$\frac{\partial n}{\partial t}=0$$

On en déduit :

$$\frac{\partial^2 n}{\partial u^2}=0$$

La densité va varier de façon linéaire à l'intérieur du tube de n_1 à n_2.

Méthode de Crank-Nicholson

La méthode étudiée précédemment est qualifiée d'explicite dans la mesure où l'équation obtenue, permettant la résolution numérique, fournissait la valeur de φ en un point en fonction des valeurs déjà connues de φ. Le problème pouvait ainsi être résolu de proche en proche.

Cependant, la condition de convergence :

$$\frac{k}{h^2}\leqslant 0{,}5$$

impliquait d'utiliser un pas temporel très petit si l'on voulait obtenir une précision spatiale correcte.

Crank et Nicholson proposèrent une méthode dite *implicite*, car elle faisait intervenir une équation de base ne fournissant pas aussi directement la valeur de φ en chaque point. L'équation de base étant toujours :

$$\frac{\partial \varphi}{\partial t}=\frac{\partial^2 \varphi}{\partial u^2}$$

la méthode diffère par le choix de l'expression utilisée pour $\dfrac{\partial^2 \varphi}{\partial u^2}$

On peut en effet décider que cette dérivée est la moyenne arithmétique des valeurs numériques qu'elle a aux temps t et $t+k$:

$$\frac{\partial^2 \varphi}{\partial u^2}=\frac{1}{2\,h^2}\,[\varphi(u+h,\ t+k)-2\,\varphi(u,\ t+k)+\varphi(u-h,\ t+k)$$
$$+\,\varphi(u+h,\ t)-2\,\varphi(u,\ t)+\varphi(u-h,\ t)]$$

D'où, pour l'équation générale $\left(\text{en posant toujours } \tau=\frac{k}{h^2} \text{ et en ayant toujours } \frac{\partial\varphi}{\partial t}=\frac{\varphi(u,\ t+k)-\varphi(u,\ t)}{k}\right)$:

$$-\,\tau\varphi(u-h,\ t+k)+(2+2\,\tau)\,\varphi(u,\ t+k)-\tau\varphi(u+h,\ t+k)=$$
$$\tau\varphi(u-h,\ t)+(2-2\,\tau)\,\varphi(u,\ t)+\tau\varphi(u+h,\ t)$$

Il est alors commode de reconsidérer le réseau maillé défini précédemment.

Prenons $t=0$. L'équation précédente, appliquée aux points d'abscisse $u=h$, $u=2\,h$, ..., $u=nh-1$, en appelant nh la *longueur* horizontale du réseau, fournit les valeurs de φ au temps $t=0+k$, en fonction de celles qui existent à $t=0$ (qui sont connues, puisque ce sont les valeurs initiales). Nous pouvons ainsi écrire $n-1$ équations, correspondant aux $(n-1)$ valeurs cherchées de φ sur l'axe Ou au temps $t=k$. Ce problème peut alors être résolu classiquement (système de $(n-1)$ équations à $(n-1)$ inconnues, voir le Chapitre 12).

Dans une seconde étape, connaissant toutes les valeurs de φ à l'instant $t=k$, il est possible de recommencer l'opération pour en déduire les nouvelles valeurs de φ à l'instant $t=k+k$, $=2\,k$, et ainsi de suite.

Cette méthode, qui semble apparemment plus longue que la précédente, a l'avantage de converger quelle que soit la valeur de $\tau=\frac{k}{h^2}$

Cependant, la précision de la méthode est liée à ce choix de τ. Une valeur telle que $\tau=$ quelques unités peut convenir.

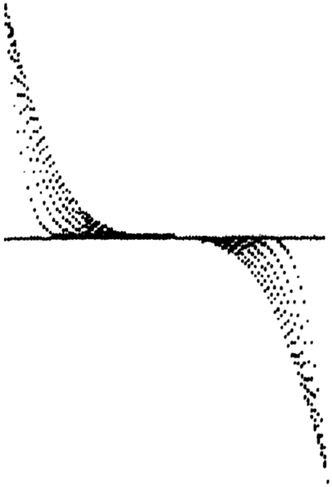
En général, on choisit $\tau=1$, car cela permet d'annuler certains coefficients de la relation établie plus haut.

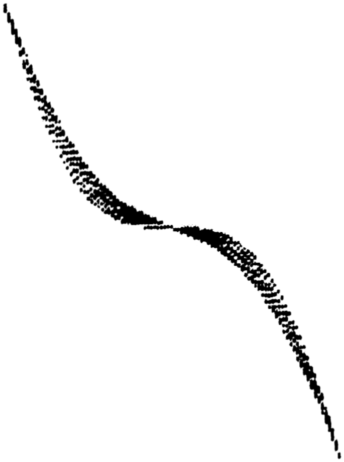
```
10 INPUT N0,N1,N2
20 INPUT"NOMBRE DE POINTS=?";N
23 INPUT"PRECISION ( 0<R<0,5)=";R
24 H=1/N:REM PAS SPATIAL
25 K=R*H^2:REM PAS TEMPOREL
26 DIMA(N):DIMB(N)
27 HIRES
30 :
40 FOR I=1 TO 99
50 A(I)=N0
55 CURSET I,199-A(I),1
60 NEXT I
70 A(0)=N1:A(100)=N2
75 CURSET 0,199-A(0),1:CURSET 100,199-A(100),1
80 :
90 FOR N = 0 TO 1000STEP 200
95 FOR U=N TO N+199
100 FOR I=1 TO 99
110 B(I)=A(I)+(K/(H^2))*(A(I+1)-2*A(I)+A(I-1))
130 NEXT I
140 :
150 FOR I=1 TO 99
160 A(I)=B(I)
170 NEXT I
180 :
183 NEXT U
184 FOR I=1TO99
185 CURSET I,199-B(I),1
186 NEXTI
187 PRINTN
190 NEXTN
```

Programme BASIC

```
10 INPUT N0,N1,N2
20 DIMA(100):DIMB(100)
23 INPUT H,K
26 HIRES
30 :
40 FOR I=1 TO 99
```

```
50 A(I)=N0
55 CURSET I,199-A(I),1
60 NEXT I
70 A(0)=N1:A(100)=N2
75 CURSET 0,199-A(0),1:CURSET 100,199-A(100),1
80 :
90 FOR N = 0 TO 1000STEP 200
95 FOR U=N TO N+199
100 FOR I=1 TO 99
110 B(I)=A(I)+(K/(H^2))*(A(I+1)-2*A(I)+A(I-1))
130 NEXT I
140 :
150 FOR I=1 TO 99
160 A(I)=B(I)
170 NEXT I
180 :
183 NEXT U
184 FOR I=1TO99
185 CURSET I,199-B(I),1
186 NEXTI
187 PRINTN
190 NEXTN
```

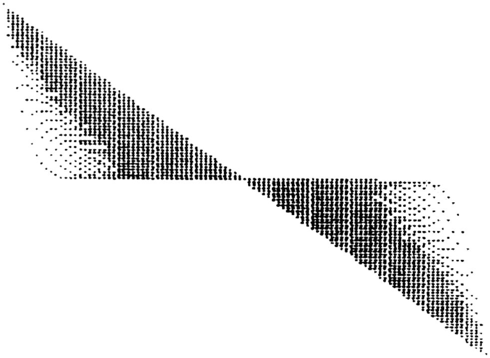




Programme Pascal

```
Run?C
ADE0    10 PROGRAM FICK2;
ADE0    20 VAR V,N,U,I,X,Y:INTEGE
R;
ADE9    30      K,H:REAL;
ADE9    40      A:ARRAY[0..100]OF
REAL;
ADE9    50      B:ARRAY[0..100]OF
REAL;
ADE9    60 (*                   *)
ADE9    70 PROCEDURE PLOT(X,Y:INT
EGER);
ADEC    80 BEGIN
AE04    90 WRITE(CHR(21),CHR(0));
AE12   100 INLINE(#FD,#21,#3A,#5C
,#DD,#46,2,#DD,#4E,4,#CD,#E5,#22
);
AE1F   105 END;
AE29   110 PROCEDURE COPY;
AE2C   112 BEGIN
AE44   114 INLINE(#FD,#21,#3A,#5C
,#FD,#CB,#01,#CE,#CD,#AC,#0E,#FD
,#CB,#01,#8E,#F3);
AE54   115 END;
AE5A   120 (*                   *)
AE5A   130 BEGIN
AE63   140 BEGIN
AE63   150 WRITE('N1');
AE70   160 READ(A[0]);
AE98   170 B[0]:=A[0];
AEDF   180 WRITE('N0');
AEEC   190 READ(A[1]);
AF14   200 WRITE('N2');
AF21   210 READ(A[100]);
AF49   220 WRITE('PAS SPATIAL');
AF5F   230 READ(H);
AF69   240 WRITE('PAS TEMPOREL');
AF80   250 READ(K);
AF8A   260 B[100]:=A[100];
AFD1   270 END;
AFD1   280 (*                   *);
AFD1   290 BEGIN
AFD1   300 WRITE(CHR(12));
AFD8   310 FOR I:=2 TO 99 DO
AFF2   320 BEGIN
AFF5   330 A[I]:=A[1];
B03C   340 END;
B03F   350 (*                   *)
;
B03F   360 FOR N:=0 TO 100 DO
B059   370 BEGIN
B05C   380 V:=N*100;
B068   390 FOR U:=V TO (V+99) DO
B08F   400 FOR I:=1 TO 99 DO
```

```
B0AC   410 BEGIN
B0AF   420 B[I]:=A[I]+(K/(H*H))*(
A[I+1]-2*A[I]+A[I-1]);
B1A0   425 A[I]:=B[I];
B1E7   430 END;
B1EE   440 FOR I:=0 TO 100 DO
B208   450 BEGIN
B20B   460 Y:=ENTIER(B[I]);
B233   470 PLOT(2*I,Y);
B248   480 END;
B24B   530 END;
B24E   540 END;
B24E   545 COPY;
B253   550 END.
End Address: B255
```



N1 = 140
N0 = 80
N2 = 20

pas spatial 0,01
pas temporel 10^{-5}

ÉQUATION HYPERBOLIQUE

C'est une équation du type :

$$\frac{\partial^2 \varphi}{\partial u^2} = \frac{\partial^2 \varphi}{\partial t^2}$$

Il s'agit d'une équation de propagation (nous la supposons ici normalisée).

En utilisant la même relation que précédemment pour exprimer numériquement ces dérivées partielles (pas spatial $= h$, pas temporel $= k$), nous pouvons écrire :

$$\frac{\varphi(u+h,\, t) - 2\,\varphi(u,\, t) + \varphi(u-h,\, t)}{h^2} = \frac{\varphi(u,\, t+k) - 2\,\varphi(u,\, t) + \varphi(u,\, t-k)}{k^2}$$

ce qui devient :

$$\varphi(u,\, t+k) = \alpha^2\,\varphi(u-h,\, t) + 2(1-\alpha^2)\,\varphi(u,\, t) + \alpha^2\,\varphi(u+h,\, t) - \varphi(u,\, t-k)$$

en posant :

$$\alpha = \frac{k}{h}$$

Cette équation peut être représentée dans un plan paramétré en u et t (voir la Figure 16.5).

Il a été démontré que l'emploi de cette méthode ne conduit à une solution convergente que si $\alpha \leqslant 1$.

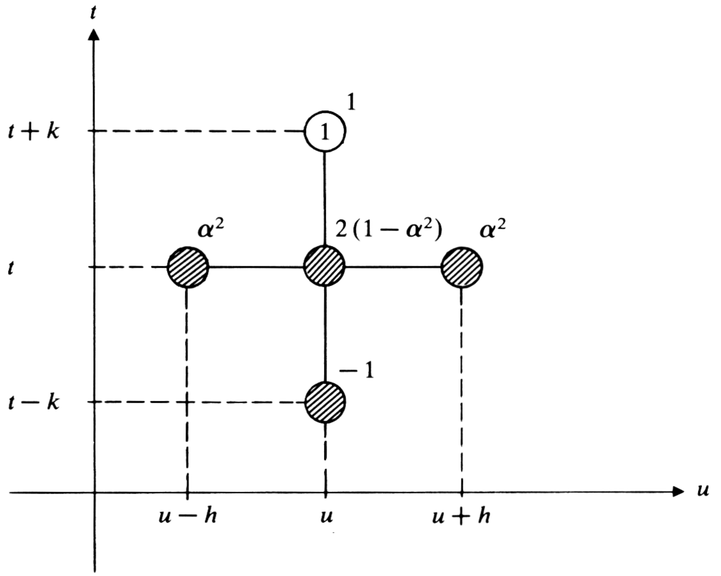


Figure 16.5.

Il existe une seconde méthode permettant de résoudre ce genre d'équations, en ramenant leur résolution à celle (simultanée) de deux équations différentielles du premier ordre.

Il suffit pour cela de poser :

$$\frac{\partial \varphi}{\partial u} = f(u, t) \quad \text{et} \quad \frac{\partial \varphi}{\partial t} = g(u, t)$$

La relation :

$$\frac{\partial^2 \varphi}{\partial u \, \partial t} = \frac{\partial^2 \varphi}{\partial t \, \partial u}$$

jointe à :

$$\frac{\partial^2 \varphi}{\partial u^2} = \frac{\partial^2 \varphi}{\partial t^2}$$

conduit alors à :

$$\begin{cases} \dfrac{\partial f(u, t)}{\partial u} = \dfrac{\partial g(u, t)}{\partial t} \\ \dfrac{\partial f(u, t)}{\partial t} = \dfrac{\partial g(u, t)}{\partial u} \end{cases}$$

Ce système d'équations différentielles peut alors se résoudre par un procédé classique.

Revenons à la première méthode et tentons de l'exploiter. Il est évident que pour commencer un calcul, il est nécessaire de connaître la valeur de la fonction φ en quatre points (valeurs initiales) pour en déduire la cinquième (et nouvelle) valeur de φ.

Considérons alors la Figure 16.6. Elle pourrait correspondre à une corde fixe à une extrémité (à droite) et excitée à gauche par un vibreur imposant une oscillation verticale sinusoïdale. Il s'agit d'un problème classique dans nos classes de terminale, connu sous le nom de *cordes*

vibrantes. Un résultat classique est le suivant : si la longueur de la corde est égale à un nombre entier de demi-longueurs d'onde, alors celle-ci est le siège d'une *résonance*, résultat de la superposition des ondes directes et réfléchies. Il est alors paradoxal de constater que, dans cette situation, la source est un *nœud* et l'amplitude de la vibration aux *ventres* est très grande. La seule différence entre ce problème réel et la simulation que nous proposons est que cette amplitude aux ventres est certainement limitée en pratique par l'extensibilité de la corde. Cependant nous constaterons que, dans notre simulation, cette amplitude ne devient pas non plus infinie, cela résultant du conflit entre la source et le nœud qu'elle devrait être!

Revenons à la Figure 16.6. Nous supposons la position des différents points (il y en a 50) contenus dans trois matrices : B(50) à l'instant $t = -2k$, A(50) à l'instant $t = -k$, C(50) à l'instant $t = 0$; la corde est ainsi supposée immobile pour $t < 0$. A partir de $t = 0$, ces données nous permettent de déduire les 50 positions des différents points, sachant que C(0) est l'excitation sinusoïdale (la source), et C(50) = 0 quel que soit t, puisqu'il s'agit de l'extrémité fixe de la corde.

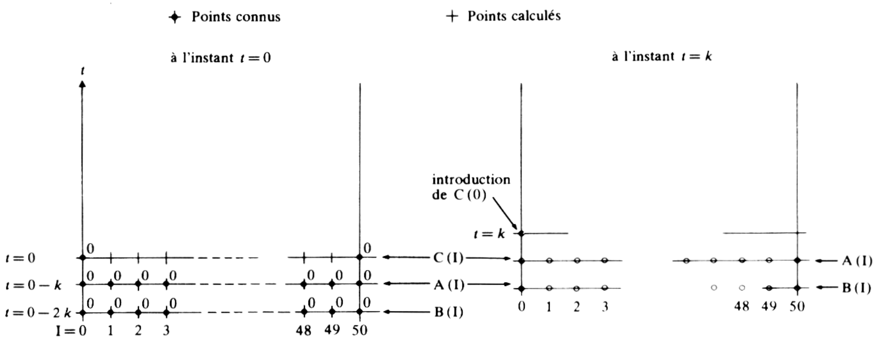


Figure 16.6.

Cela étant fait, nous pouvons passer à l'instant $t = k$.

Les deux positions « anciennes » dont nous avons besoin pour effectuer les calculs sont par définition A (50) et B (50).

Or, ce sont justement les matrices C (50) et A (50) évaluées à l'instant $t = 0$ et $t = -k$. Il suffit d'effectuer le transfert des coefficients des unes vers les autres pour pouvoir effectuer une nouvelle itération; d'où le programme :

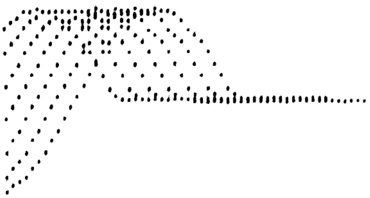
```
1 INPUT"PRECISION (0<A<1) = ";A
2 LET H=1/50:K=A*H
3 HIRES
10 DIMA(50):DIMB(50):DIMC(50)
20 FOR I=0 TO 50:B(I)=0:NEXT
30 FOR I=1 TO 50:A(I)=0:NEXT
40 C(0)=SIN(2*PI*T):C(50)=0
50 FOR I=1 TO 49
60 C(I)=A*A*A(I-1)+2*(1-A*A)*A(I)+A*A*A
(I+1)-B(I)
70 CURSETI,100-99*C(I),1
80 NEXTI
90 FOR I=0 TO 50
100 B(I)=A(I)
110 A(I)=C(I)
120 NEXTI
130 T=T+K
140 HIRES:GOTO40
```

Il ne faut pas oublier, comme pour tous les problèmes de ce type, qu'il y a au départ un régime transitoire qui, suivant les conditions initiales, peut être très long.

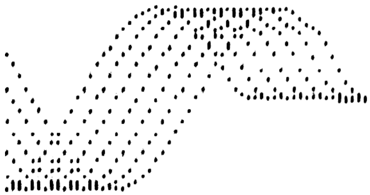
A = 0,5
N = 57
K = 0,01



① $0 \leqslant t \leqslant 0{,}36$



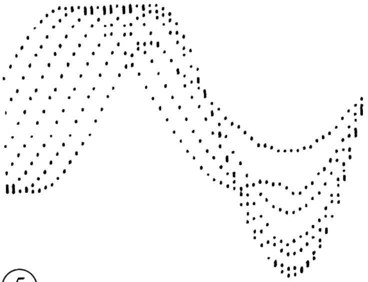
② $0{,}36 \leqslant t \leqslant 0{,}72$

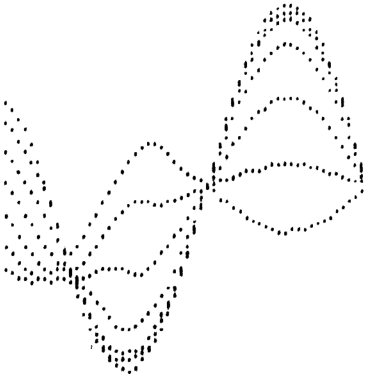


③ 1, $0{,}72 \leqslant t \leqslant 1{,}08$

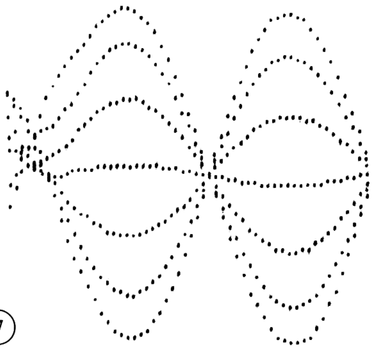


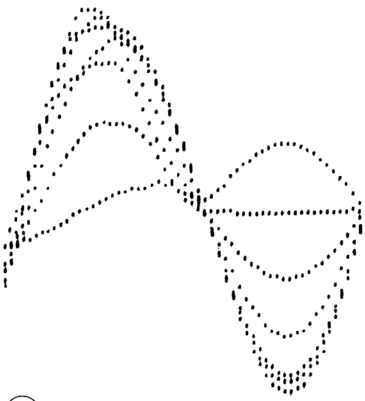
④ 22


(5)

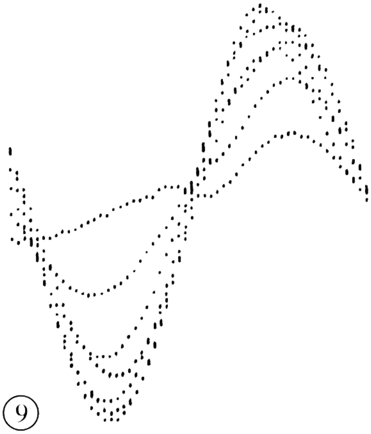

(6)

A = 0,5
N = 57
K = 0,01


(7)



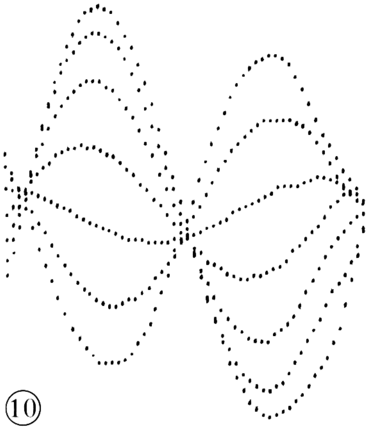
(8)



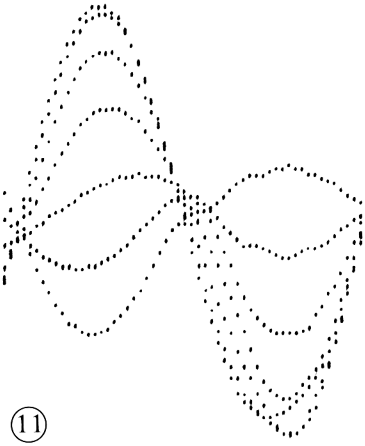
(9)

A = 0,5

N = 57
K = 0,01
(de (1) à (11) : 396 itérations)



(10)

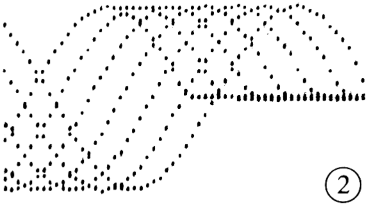


(11)

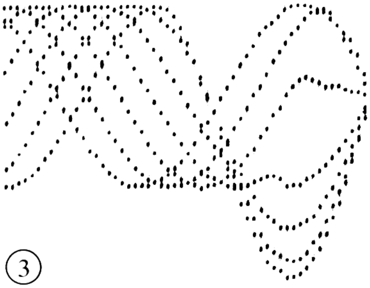


①

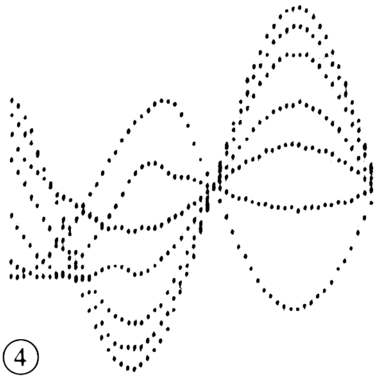
A = 0,8
K = 0,016
N = 57
(de ① à ⑧ : 288 itérations)



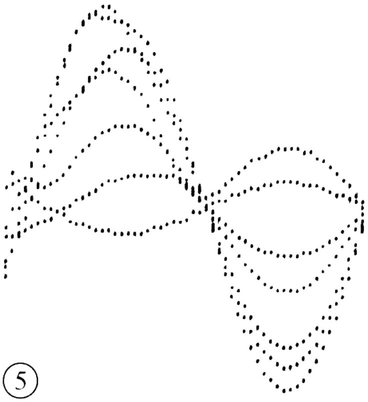
②



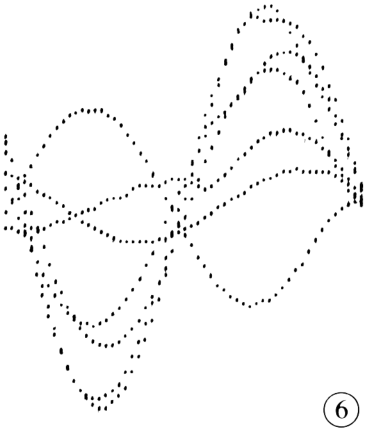
③



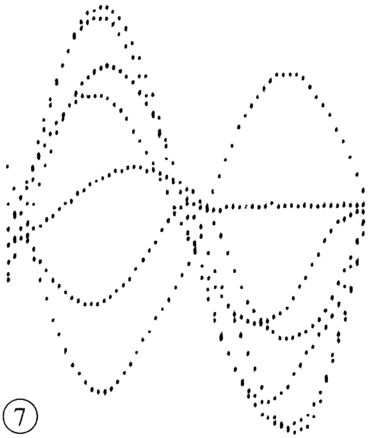
④



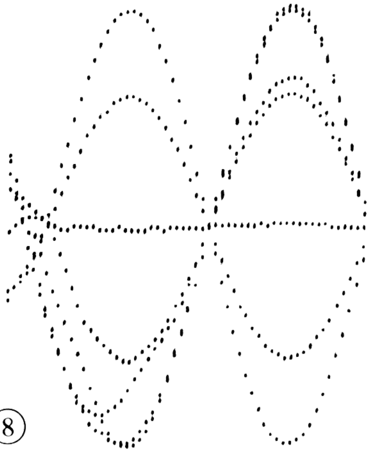
⑤



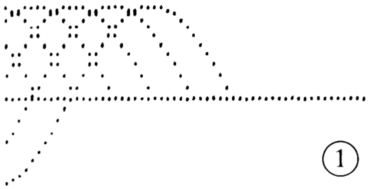
⑥



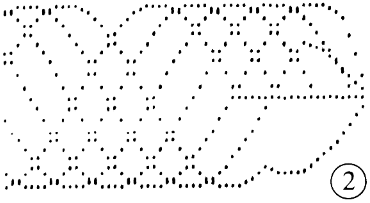
⑦



⑧

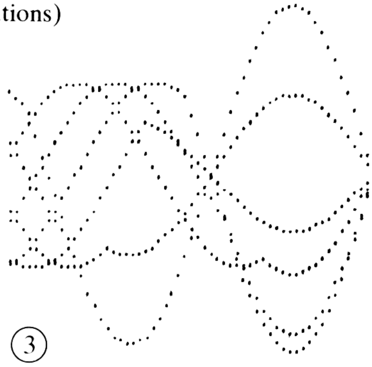


①

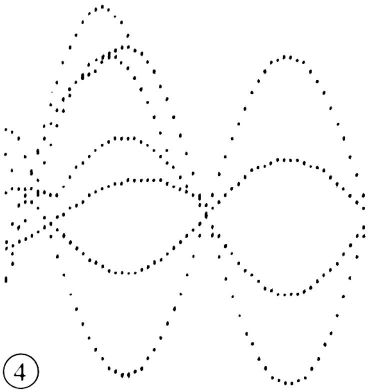


②

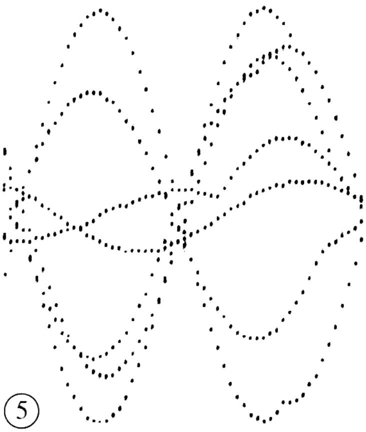
A = 1
K = 0,02
N = 57
(de ① à ⑧ : 288 itérations)



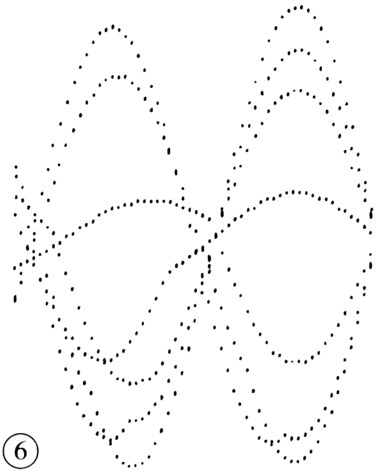
③



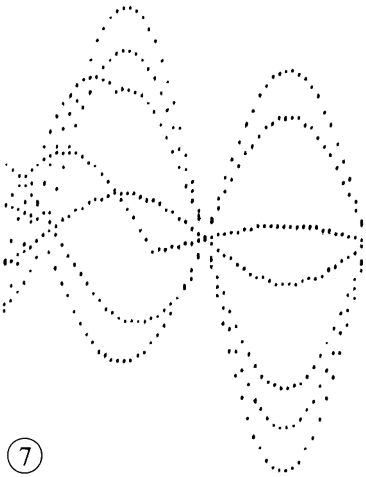
④



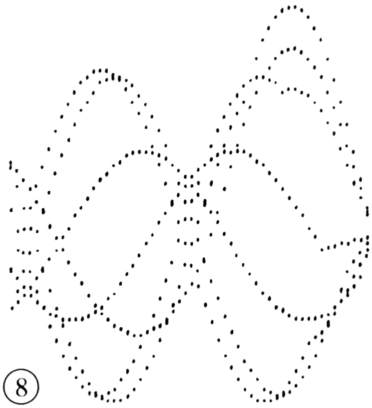
(5)



(6)



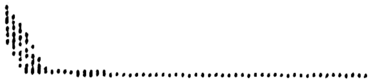
(7)



(8)



(1)



(2)

A = 0,1
K = 0,02
N = 57
(de (1) à (8) : 288 itérations)



(3)



(4)



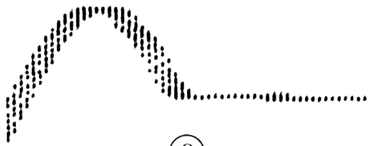
⑤



⑥



⑦

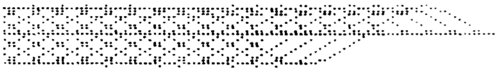


⑧

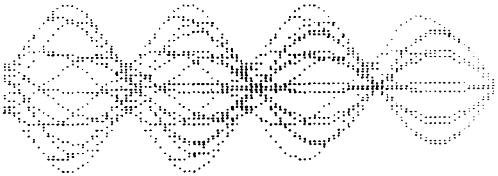
```
Run?C
ADA8    10 PROGRAM HYPERBOLE;
ADA8    20 LABEL 100;
ADAE    30 CONST H=0.02;
ADAE    35       N=103;
ADAE    40       PI=3.14159;
ADAE    50       E=0.5;
ADAE    60       K=0.01;
ADAE    70 VAR I:INTEGER;
ADB7    80     X,Y:INTEGER;
ADB7    90     T,F:REAL;
ADB7   100     A:ARRAY[0..N]OF RE
AL;
ADB7   110     B:ARRAY[0..N]OF RE
AL;
ADB7   120     C:ARRAY[0..N]OF RE
AL;
ADB7   121 PROCEDURE COPY;
ADBA   122 BEGIN
ADD2   123 INLINE(#FD,#21,#3A,#5C
,#FD,#CB,#01,#CE,#CD,#AC,#0E,#FD
,#CB,#01,#8E,#F3);
ADE2   129 END;
ADE8   130 PROCEDURE PLOT(X,Y:INT
EGER);
ADEB   140 BEGIN
AE03   150 WRITE(CHR(21),CHR(0));
AE11   160 INLINE(#FD,#21,#3A,#5C
,#DD,#46,2,#DD,#4E,4,#CD,#E5,#22
);
AE1E   170 END;
AE28   180 BEGIN
AE31   185 F:=0;
AE3E   190 T:=0;
AE4B   195 PAGE;
AE50   200 FOR I:=0 TO N DO B[I]:
=0;
AE9B   210 FOR I:=0 TO N DO A[I]:
=0;
AEE6   220 100:C[0]:=SIN(2*PI*T);
AF2B   230 C[N]:=0;
AF56   240 FOR I:=1 TO (N-1)  DO
AF75   250 BEGIN{$C-}
AF78   260 C[I]:=(E*E*A[I-1])+(2*
(1-E*E)*A[I])+((E*E)*A[I+1])-B[I
];
B096   261 IF ABS(C[I]) < 1E-15 T
HEN C[I]:=0;
B0FC   270 X:=2*I;
B10A   280 Y:=ENTIER(100+10*C[I])
;
B148   290 IF ENTIER(F/10)=(F/10)
 THEN PLOT(X,Y);
B194   300 END;
B198   310 FOR I:=0 TO N DO B[I]:
=A[I];
```

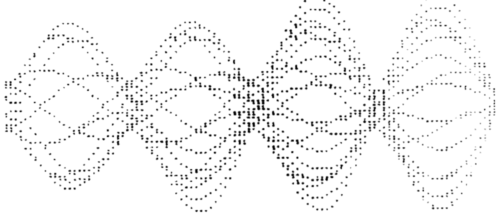
```
B1FC  320 FOR I:=0 TO N DO A[I]:
=C[I];
B260  330 T:=T+K;
B279  345 IF F=200 THEN COPY;
B29F  346 IF F=200 THEN PAGE;
B2C5  347 IF F=200 THEN F:=0;
B2F3  348 F:=F+1;
B30C  350 GOTO 100;
B30F  360 END.
End Address: B311
```

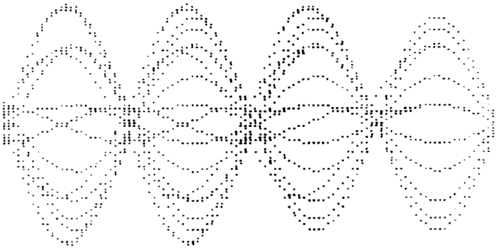
N = 103
M = 0,02
K = 0,01

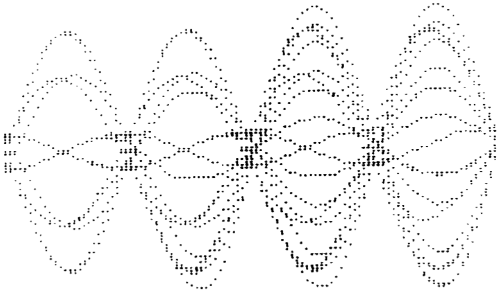


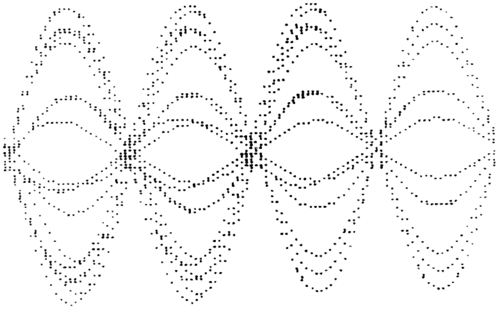


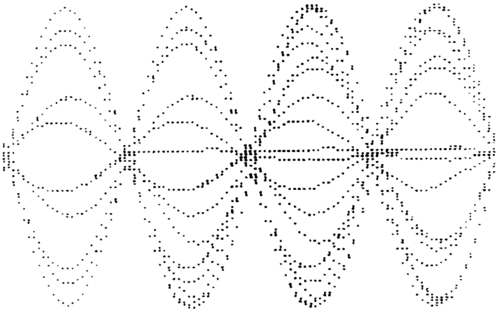












ÉQUATION ELLIPTIQUE

C'est une équation de la forme :

$$\frac{\partial^2 \varphi}{\partial u^2}+\frac{\partial^2 \varphi}{\partial y^2}=0$$

connue en physique sous le nom d'équation de Poisson. Plus généralement, on peut l'écrire :

$$\frac{\partial^2 \varphi}{\partial u^2}+\frac{\partial^2 \varphi}{\partial y^2}=f(u, y)$$

En utilisant la même relation que précédemment pour exprimer numériquement ces dérivées partielles (nous supposons la maille *carrée*, c'est-à-dire que le pas sur l'axe Ou, soit h, est égal au pas sur l'axe Oy (voir la Figure 16.7)) :

$$0=\varphi(u+h, y)-2\varphi(u, y)+\varphi(u-h-y)+\varphi(u, y+h)-2\varphi(u, y)$$
$$+\varphi(u, y-h)$$

D'où :

$$4\varphi(u, y)-\varphi(u+h, y)-\varphi(u-h, y)-\varphi(u, y+h)-\varphi(u, y-h)=0$$

Considérons par exemple une plaque rectangulaire, pour laquelle la température des bords est imposée par un certain dispositif et sur laquelle nous disposons un certain nombre de points de mesure équidistants.

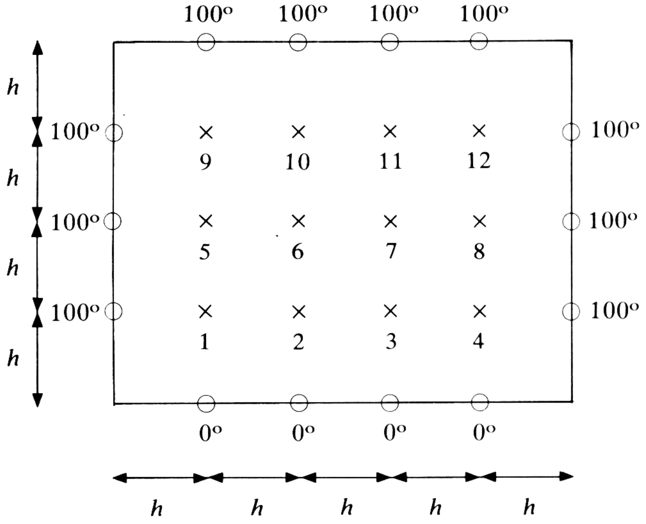


Figure 16.7.

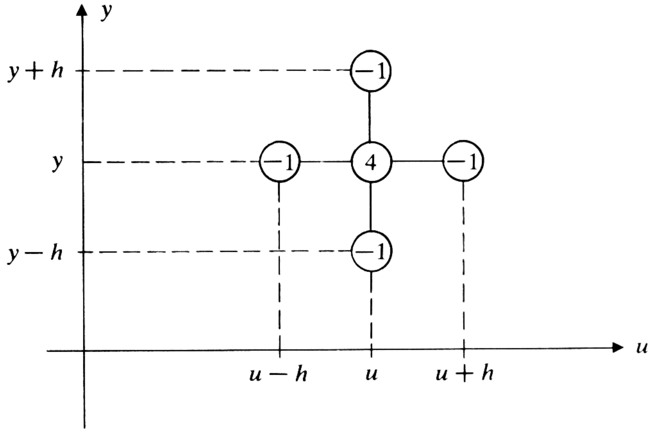


Figure 16.8.

La relation précédente impose une relation entre certains groupements de points. On peut symboliser cela par le dessin suivant (Figure 16.8), qui est une représentation spatiale de cette relation. A chaque point d'abscisse u et d'ordonnée y est associée une valeur de la fonction $\varphi(u, y)$, elle-même reliée aux valeurs de φ pour les points entourant celui qui est considéré.

Revenons à la plaque rectangulaire définie précédemment. Supposons que tous les bords soient maintenus à la température de 100 °C, sauf le bord inférieur qui, lui, est porté à 0 °C. Intéressons-nous à la température de chaque point intérieur [1].

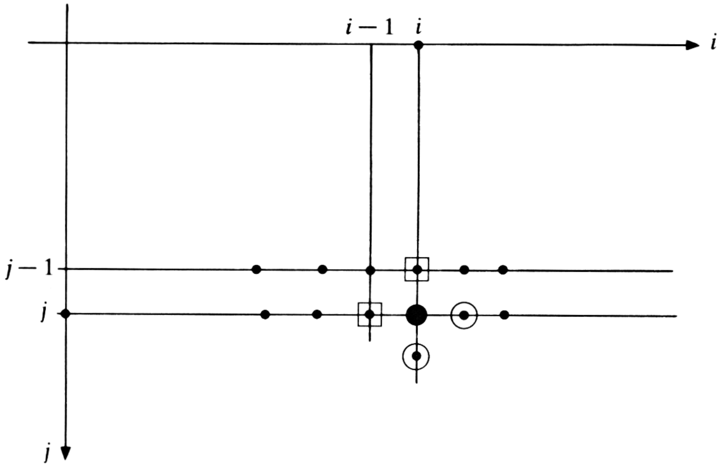


Figure 16.9.

Considérons le point (1). La valeur de la fonction φ en ce point, soit φ_1, est telle que :

$$4\,\varphi_1 - 1 \times (100) - 1 \times (\varphi_5) - 1 \times (\varphi_2) - 1 \times (0) = 0$$

---

1. D. Herrmann : *40 BASIC Programme*, Vieweg, Wiesbaden, 1983.

G. D. Smith : *Numerical Solution of Partial Differential Equations*, Oxford University Press.

soit, en ordonnant :

$$4\,\varphi_1 - 1 \times \varphi_2 + 0 \times \varphi_3 + 0 \times \varphi_4 - 1 \times \varphi_5 + \ldots = 1 \times (100) + 1 \times 0$$
$$= 1 \times (100 + 0)$$

Continuons; pour le point (2), nous avons :

$$4\,\varphi_2 - \varphi_1 - \varphi_6 - \varphi_3 - 0 = 0$$

D'où :

$$-1 \times \varphi_1 + 4 \times \varphi_2 - 1 \times \varphi_3 \ldots - 1 \times \varphi_6 + \ldots = 1 \times (0)$$

On conçoit donc qu'à chaque point intérieur sont associées :

- une série de coefficients;
- une température de « bord » égale à la plus proche considérée, sauf pour les angles ((1), (4), (9), (12)), auxquels on associe la somme des températures des bords de coins les plus proches.

Ainsi :

$u_1 \longrightarrow 0 + 100$
$u_2 \longrightarrow 0$
$u_3 \longrightarrow 0$
$u_4 \longrightarrow 0 + 100$
$u_5 \longrightarrow 100$
.
.
.
$u_9 \longrightarrow 100 + 100$
.
.
.
$u_{12} \longrightarrow 100 + 100$

On peut ainsi écrire, en tenant compte du fait qu'à chaque point du plan est associée une ligne de coefficients :

$$\begin{pmatrix} 100 \\ 0 \\ 0 \\ 100 \\ \cdot \\ \cdot \\ \cdot \\ 200 \\ 0 \\ 0 \\ 200 \end{pmatrix} = \begin{pmatrix} 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \varphi_4 \\ \cdot \\ \cdot \\ \cdot \\ \varphi_9 \\ \varphi_{10} \\ \varphi_{11} \\ \varphi_{12} \end{pmatrix}$$

Le problème revient donc à trouver la solution de ce système d'équations.

Il existe alors deux types de méthodes applicables à la résolution d'un tel système :

- la méthode directe (résolution du système d'équations, qui peut être très longue; voir le chapitre concerné);
- les méthodes itératives qui, partant d'un ensemble de valeurs φ_i assez quelconques, conduisent après plusieurs passages à une convergence éventuelle vers la solution.

Nous décrivons ici deux de ces méthodes :

Liebmann non extrapolée ou méthode de Gauss-Seidel (méthode de relaxation)

Considérons le $(n+1)^{\text{ième}}$ passage; nous pouvons écrire :

$$\varphi_{i,j}^{(n+1)} = \frac{1}{4}\left[\varphi_{i-1,j}^{(n+1)} + \varphi_{i+1,j}^{(n)} + \varphi_{i,j-1}^{(n+1)} + \varphi_{i,j+1}^{(n)}\right]$$

Cette équation permet d'évaluer au $(n+1)^{\text{ième}}$ passage $\varphi_{i,j}$, connaissant :

- $\varphi_{i+1,j}$ et $\varphi_{i,j+1}$ évalués au passage n précédent (points cerclés sur le dessin);
- $\varphi_{i,j-1}$ et $\varphi_{i-1,j}$ évalués pendant ce même passage $n+1$ (points entourés d'un carré). (Voir la Figure 16.9.)

$\varphi_{i,j}$ est aussi estimée à l'aide de valeurs de φ les plus récemment connues.

Il a été montré que cette méthode converge.

On peut évaluer la précision à la $(n+1)^{\text{ième}}$ itération, en calculant le résidu $R_{i,j}$ au point (i, j) :

$$R_{i,j} = \varphi_{i-1,j}^{(n+1)} + \varphi_{i+1,j}^{(n)} + \varphi_{i,j-1}^{(n+1)} + \varphi_{i,j+1}^{(n)} - 4\,\varphi_{i,j}^{(n)}$$

quantité qui est évidemment nulle pour *la* solution.

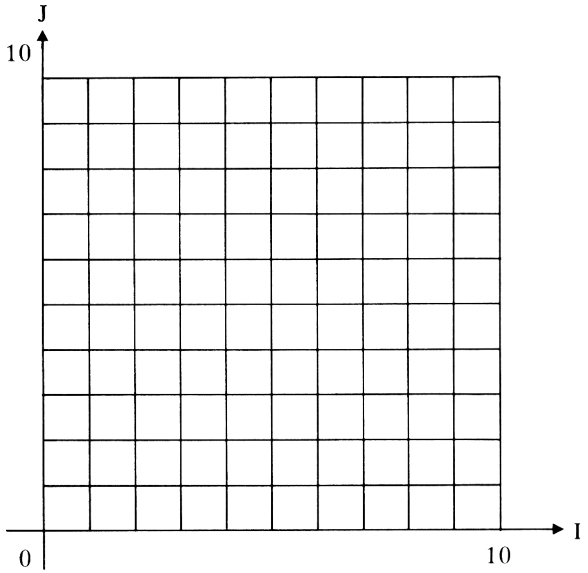


Figure 16.10.

Exemple

Considérons une fonction φ à deux variables, qui obéit à l'équation :

$$\frac{\partial^2 \varphi}{\partial u^2}+\frac{\partial^2 \varphi}{\partial y^2}=0$$

Dans un réseau rectangulaire (voir la Figure 16.10), cette fonction prend les valeurs : 50 pour les bords gauche et droit, 100 pour le bord supérieur, 10 pour le bord inférieur.

La matrice P (I, J) contient à chaque instant les valeurs de la fonction :

$$\varphi^{(n+1)}\,(\mathrm{I},\,\mathrm{J})$$

Q (I, J) contient les valeurs antérieures $\varphi^{(n)}$ (I, J). A la fin de chaque itération, Q est remplie avec les éléments de P.

Les résultats (valeur de φ et des résidus) sont imprimés tous les cinq passages.

Pour commencer le calcul, il faut donner des valeurs arbitraires initiales aux éléments de Q. Ici, nous avons pris Q (I, J) = 50, ce qui semble raisonnable. On aurait pu également faire un calcul préalable, en n'évaluant la fonction φ que pour un point sur deux (16 points à calculer au lieu de 81).

```
 5 DIM P(10,10):DIM Q(10,10): DIM R(10,1
0):N=0
 10 REM-----------------------
 20 REM CONDITIONS AUX LIMITES
 30 REM-----------------------
 40 FOR I=0 TO 10
 50 P(I,0)=10
 55 Q(I,0)=10
 60 NEXT
 70 FOR I=0 TO 10
 80 P(I,10)=100
 90 Q(I,10)=100
 95 NEXT
 100 FOR J = 0 TO 10
 110 P(0,J)=50
```

```
 115 Q(0,J)=50
 120 NEXT
 130 FOR J=0 TO 10
 140 P(10,J)=50
 150 Q(10,J)=50
 160 NEXT
 200 REM------------------------
 210 REM ETAT INITIAL QUELCONQUE
 220 REM------------------------
 230 FOR I=1 TO 9
 240 FOR J=1 TO 9
 250 Q(I,J)=50
 260 NEXT
 270 NEXT
 300 REM--------
 310 REM CALCULS
 320 REM--------
 330 FOR J=1 TO 9
 340 FOR I=1 TO 9
 350 P(I,J)=.25*(P(I-1,J)+Q(I+1,J)+P(I,J
-1)+Q(I,J+1))
 360 NEXT I
 370 NEXT J
 380 N=N+1
 390 PRINTN
 400 REM---------------------
 410 REM EVALUATION DU RESIDU
 420 REM---------------------
 430 I0=INT(RND(1)*9+1)
 440 J0=INT(RND(1)*9+1)
 450 R=P(I0-1,J0)+Q(I0+1,J0)+P(I0,J0-1)+
Q(I0,J0+1)-4*Q(I0,J0)
 460 RS=R/P(I0,J0)
 470 PRINTRS
 500 REM----------
 510 REM ITERATION
 520 REM----------
 530 FOR I=1 TO 9
 540 FOR J=1 TO 9
 545 R(I,J)=P(I-1,J)+Q(I+1,J)+P(I,J-1)+Q
(I,J+1)-4*Q(I,J)
 550 Q(I,J)=P(I,J)
```

```
560 NEXT J
570 NEXT I
580 IF N=1 THEN GOSUB 1100
585 IF INT (N/5)=N/5 THEN GOSUB 1100
590 GOTO 300
1100 FOR J=10 TO 0 STEP -1
1110 FOR I=0 TO 10
1120 LPRINT TAB(I*6);INT(R(I,J)*10)/10;
1130 NEXT I
1140 LPRINT
1150 NEXT J
1160 LPRINT:LPRINT"Passage numero ";N
1165 LPRINT:LPRINT:LPRINT:LPRINT
1170 RETURN
```

0	0	0	0	0	0	0	0	0	0	0
0	49.9	62.4	65.6	66.4	66.5	66.6	66.6	66.6	66.6	0
0	-.1	-.1	-.1	-.1	-.1	-.1	-.1	-.1	-.1	0
0	-.1	-.1	-.1	-.1	-.1	-.1	-.1	-.1	-.1	0
0	-.1	-.1	-.2	-.2	-.3	-.3	-.3	-.3	-.3	0
0	-.2	-.4	-.5	-.6	-.7	-.7	-.7	-.7	-.7	0
0	-.7	-1.3	-1.7	-1.9	-2	-2	-2	-2	-2	0
0	-2.5	-4.4	-5.4	-5.8	-5.9	-6	-6	-6	-6	0
0	-10	-15	-16.9	-17.5	-17.7	-17.8	-17.8	-17.8	-17.8	0
0	-40	-50	-52.5	-53.2	-53.3	-53.4	-53.4	-53.4	-53.4	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 1

0	0	0	0	0	0	0	0	0	0	0
0	3.2	5.1	6.2	6.8	7.1	6.9	6	4.1	1.9	0
0	5	8.1	9.8	10.7	11.2	11.4	10.3	7.5	3.5	0
0	3.7	6.2	7.7	8.4	8.8	8.9	9.1	7.3	3.7	0
0	1	1.9	2.5	2.7	2.8	2.8	3	3.3	2.1	0
0	-.6	-1	-1.4	-1.6	-1.8	-1.8	-1.6	-1	-.2	0
0	-1.5	-2.6	-3.4	-3.9	-4.2	-4.3	-3.8	-2.8	-1.5	0
0	-2.3	-4	-5.1	-5.8	-6.1	-6.1	-5.4	-3.9	-2	0
0	-3	-5	-6.2	-6.9	-7.3	-7.2	-6.3	-4.4	-2.2	0
0	-2.7	-4.2	-5.1	-5.6	-5.9	-5.8	-5	-3.5	-1.7	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 5

0	0	0	0	0	0	0	0	0	0	0
0	.7	1.2	1.5	1.7	1.7	1.5	1.2	.8	.4	0
0	1.2	2.2	2.8	3.2	3.2	2.9	2.4	1.6	.7	0
0	1.4	2.4	3.2	3.6	3.7	3.5	2.8	1.9	.9	0
0	1	1.8	2.4	2.7	2.9	2.8	2.4	1.7	.8	0
0	.3	.6	.8	1	1.1	1.2	1.2	.9	.5	0
0	-.4	-.7	-.9	-.9	-.8	-.6	-.3	-.1	0	0
0	-.9	-1.5	-1.9	-2.2	-2.1	-1.9	-1.4	-.9	-.4	0
0	-1	-1.7	-2.1	-2.4	-2.4	-2.2	-1.7	-1.2	-.6	0
0	-.7	-1.1	-1.5	-1.6	-1.7	-1.5	-1.2	-.8	-.4	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 10

0	0	0	0	0	0	0	0	0	0	0
0	.2	.4	.5	.6	.6	.5	.4	.2	.1	0
0	.4	.8	1	1.1	1.1	1	.8	.5	.2	0
0	.5	1	1.3	1.4	1.4	1.3	1	.7	.3	0
0	.5	.9	1.2	1.4	1.4	1.3	1	.7	.3	0
0	.3	.6	.8	.9	1	.9	.8	.5	.3	0
0	0	.1	.2	.3	.4	.4	.4	.3	.1	0
0	-.2	-.3	-.3	-.3	-.2	-.1	0	0	0	0
0	-.3	-.5	-.6	-.6	-.5	-.4	-.3	-.2	-.1	0
0	-.2	-.4	-.5	-.5	-.5	-.4	-.3	-.2	-.1	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 15

0	0	0	0	0	0	0	0	0	0	0
0	.1	.1	.2	.2	.2	.2	.1	.1	0	0
0	.1	.3	.4	.5	.5	.4	.3	.2	.1	0
0	.2	.4	.6	.6	.6	.6	.4	.3	.1	0
0	.2	.4	.6	.7	.7	.6	.5	.3	.1	0
0	.2	.4	.5	.6	.6	.6	.5	.3	.1	0
0	.1	.2	.3	.4	.4	.4	.3	.2	.1	0
0	0	.1	.1	.2	.2	.2	.2	.1	0	0
0	-.1	0	0	0	0	.1	.1	0	0	0
0	-.1	-.1	-.1	-.1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 20

0	0	0	0	0	0	0	0	0	0	0
0	0	0	.1	.1	.1	.1	0	0	0	0
0	0	.1	.2	.2	.2	.2	.1	.1	0	0
0	.1	.2	.3	.3	.3	.3	.2	.1	0	0
0	.1	.2	.3	.3	.3	.3	.2	.2	.1	0
0	.1	.2	.3	.3	.4	.3	.2	.2	.1	0
0	.1	.2	.3	.3	.3	.3	.2	.1	0	0
0	0	.1	.2	.2	.2	.2	.2	.1	0	0
0	0	0	.1	.1	.1	.1	.1	.1	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 25

```
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     .1    .1    .1    .1    0     0     0     0
0     0     .1    .1    .1    .1    .1    .1    0     0     0
0     0     .1    .2    .2    .2    .2    .1    .1    0     0
0     0     .1    .2    .2    .2    .2    .1    .1    0     0
0     0     .1    .2    .2    .2    .2    .1    .1    0     0
0     0     .1    .1    .1    .1    .1    .1    .1    0     0
0     0     0     .1    .1    .1    .1    .1    0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0

Passage numero  30


0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     .1    .1    .1    .1    0     0     0     0
0     0     0     .1    .1    .1    .1    0     0     0     0
0     0     0     .1    .1    .1    .1    .1    0     0     0
0     0     0     .1    .1    .1    .1    .1    0     0     0
0     0     0     .1    .1    .1    .1    0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0

Passage numero  35


0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0

Passage numero  40
```

Remarque

Ce changement de la ligne 1120 permet d'imprimer les valeurs de :

```
1120 LPRINT TAB(I*6);INT(P(I,J)*10)/10;
```

50	100	100	100	100	100	100	100	100	100	50
50	62.4	65.6	66.4	66.6	66.6	66.6	66.6	66.6	66.6	50
50	49.9	49.9	49.9	49.9	49.9	49.9	49.9	49.9	49.9	50
50	49.9	49.9	49.9	49.9	49.9	49.9	49.9	49.9	49.9	50
50	49.9	49.9	49.9	49.9	49.9	49.9	49.9	49.9	49.9	50
50	49.9	49.9	49.8	49.8	49.8	49.8	49.8	49.8	49.8	50
50	49.8	49.6	49.5	49.5	49.5	49.5	49.5	49.5	49.5	50
50	49.3	48.9	48.6	48.5	48.5	48.5	48.5	48.5	48.5	50
50	47.5	46.2	45.7	45.6	45.5	45.5	45.5	45.5	45.5	50
50	40	37.5	36.8	36.7	36.6	36.6	36.6	36.6	36.6	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 1

50	100	100	100	100	100	100	100	100	100	50
50	71.5	78.6	81.1	82	82.4	82.4	82	79.8	72.5	50
50	58.6	63.4	65.5	66.4	66.8	66.9	66.7	65.2	60.3	50
50	52.6	54.5	55.5	55.9	56.1	56.2	56.2	55.8	54	50
50	50.2	50.5	50.6	50.6	50.6	50.5	50.6	50.7	50.7	50
50	49.3	48.8	48.4	48.1	47.9	47.9	47.9	48.3	49	50
50	48.4	47.1	46.3	45.8	45.5	45.4	45.5	46.2	47.7	50
50	46.5	44	42.5	41.8	41.4	41.3	41.6	42.7	45.4	50
50	42.2	37.9	35.8	34.9	34.5	34.4	34.8	36.5	41.1	50
50	32.7	27	25	24.2	24	23.9	24.3	26	31.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 5

50	100	100	100	100	100	100	100	100	100	50
50	73.1	81.3	84.6	85.9	86.3	86.1	85	81.8	73.5	50
50	61.5	68.1	71.5	73.2	73.8	73.6	72.3	69	62.1	50
50	55.4	59.2	61.5	62.7	63.2	63.1	62.3	60.2	56.1	50
50	51.8	53.2	54.1	54.6	54.9	54.9	54.7	53.9	52.3	50
50	49.4	49	48.7	48.5	48.4	48.5	48.8	49.2	49.6	50
50	47.4	45.4	44	43.2	42.8	42.9	43.7	45.1	47.2	50
50	44.8	41.1	38.7	37.4	36.9	37.1	38.1	40.4	44.3	50
50	40.3	34.7	31.7	30.2	29.6	29.8	31	34	39.8	50
50	31.4	24.7	22	20.9	20.5	20.7	21.6	24.2	31	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 10

50	100	100	100	100	100	100	100	100	100	50
50	73.6	82.2	85.7	87.1	87.5	87.2	85.8	82.4	73.7	50
50	62.3	69.7	73.6	75.4	76	75.6	73.9	70.1	62.6	50
50	56.4	61.1	63.9	65.4	65.9	65.7	64.3	61.6	56.8	50
50	52.7	54.8	56.2	57	57.3	57.2	56.6	55.3	53	50
50	49.9	49.8	49.8	49.8	49.9	50	50.1	50.1	50.1	50
50	47.3	45.3	43.9	43.2	43	43.2	44	45.4	47.4	50
50	44.3	40.3	37.7	36.4	36	36.4	37.6	40.2	44.3	50
50	39.7	33.6	30.4	28.8	28.3	28.7	30.2	33.4	39.6	50
50	30.9	24	21.1	19.9	19.5	19.8	21	23.8	30.8	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 15

50	100	100	100	100	100	100	100	100	100	50
50	73.8	82.5	86.1	87.6	88	87.6	86.2	82.6	73.8	50
50	62.7	70.3	74.4	76.3	76.9	76.4	74.5	70.5	62.8	50
50	56.9	61.9	65	66.6	67.1	66.7	65.2	62.1	57	50
50	53.1	55.6	57.3	58.2	58.6	58.4	57.5	55.9	53.3	50
50	50.2	50.4	50.6	50.8	50.9	50.9	50.9	50.7	50.4	50
50	47.5	45.6	44.4	43.8	43.6	43.9	44.5	45.8	47.6	50
50	44.3	40.3	37.8	36.6	36.2	36.6	37.9	40.4	44.4	50
50	39.6	33.5	30.2	28.6	28.2	28.7	30.2	33.5	39.7	50
50	30.8	23.8	20.9	19.7	19.4	19.7	20.9	23.8	30.8	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 20

50	100	100	100	100	100	100	100	100	100	50
50	73.9	82.7	86.3	87.8	88.2	87.8	86.3	82.7	73.9	50
50	62.9	70.6	74.8	76.7	77.3	76.8	74.8	70.7	62.9	50
50	57.1	62.3	65.5	67.1	67.7	67.2	65.6	62.4	57.2	50
50	53.4	56	57.8	58.8	59.2	58.9	58	56.2	53.5	50
50	50.4	50.9	51.2	51.4	51.5	51.5	51.3	51	50.5	50
50	47.7	45.9	44.8	44.3	44.1	44.3	44.9	46.1	47.8	50
50	44.4	40.5	38.1	36.9	36.6	37	38.2	40.6	44.5	50
50	39.7	33.6	30.3	28.8	28.4	28.9	30.4	33.6	39.7	50
50	30.8	23.8	21	19.8	19.5	19.8	21	23.8	30.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 25

50	100	100	100	100	100	100	100	100	100	50
50	73.9	82.7	86.4	87.9	88.3	87.9	86.4	82.8	73.9	50
50	62.9	70.8	74.9	76.9	77.6	77	75	70.8	63	50
50	57.2	62.5	65.7	67.5	68	67.5	65.8	62.6	57.3	50
50	53.5	56.3	58.2	59.2	59.6	59.3	58.2	56.4	53.6	50
50	50.6	51.1	51.5	51.8	51.9	51.8	51.6	51.2	50.6	50
50	47.8	46.1	45.1	44.6	44.5	44.7	45.2	46.2	47.8	50
50	44.5	40.6	38.3	37.2	36.8	37.2	38.4	40.7	44.6	50
50	39.7	33.7	30.5	29	28.6	29	30.5	33.7	39.8	50
50	30.9	23.9	21	19.9	19.6	19.9	21.1	23.9	30.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 30

50	100	100	100	100	100	100	100	100	100	50
50	73.9	82.8	86.4	88	88.4	88	86.5	82.8	73.9	50
50	63	70.9	75.1	77.1	77.7	77.1	75.1	70.9	63	50
50	57.3	62.6	65.9	67.6	68.2	67.7	65.9	62.7	57.3	50
50	53.6	56.4	58.4	59.4	59.8	59.5	58.4	56.5	53.6	50
50	50.6	51.3	51.7	52	52.1	52	51.8	51.3	50.7	50
50	47.9	46.3	45.3	44.8	44.7	44.8	45.4	46.3	47.9	50
50	44.6	40.8	38.5	37.4	37	37.4	38.5	40.8	44.6	50
50	39.8	33.8	30.6	29.2	28.7	29.2	30.7	33.8	39.8	50
50	30.9	23.9	21.1	20	19.6	20	21.1	23.9	30.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 35

50	100	100	100	100	100	100	100	100	100	50
50	73.9	82.8	86.5	88	88.4	88	86.5	82.8	73.9	50
50	63	70.9	75.1	77.1	77.7	77.2	75.1	70.9	63	50
50	57.3	62.7	66	67.7	68.3	67.7	66	62.7	57.3	50
50	53.6	56.5	58.5	59.5	59.9	59.6	58.5	56.6	53.6	50
50	50.7	51.3	51.8	52.1	52.2	52.2	51.9	51.4	50.7	50
50	47.9	46.4	45.4	45	44.8	45	45.5	46.4	47.9	50
50	44.6	40.8	38.6	37.5	37.1	37.5	38.6	40.9	44.7	50
50	39.8	33.8	30.7	29.2	28.8	29.3	30.7	33.8	39.8	50
50	30.9	24	21.1	20	19.7	20	21.2	24	30.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 40

Liebmann extrapolée ou S.O.R. (successive over relaxation)

L'itération précédente peut aussi s'écrire :

$$\varphi_{i,j}^{(n+1)} = \varphi_{i,j}^{(n)} + \frac{1}{4}\left[\varphi_{i-1,j}^{(n+1)} + \varphi_{i+1,j}^{(n)} + \varphi_{i,j-1}^{(n+1)} + \varphi_{i,j+1}^{(n)} - 4\,\varphi_{i,j}^{(n)}\right]$$

Mais le terme entre crochets est précisément ce que nous avons appelé précédemment $R_{i,j}$, résidu au point i, j :

$$\varphi_{i,j}^{(n+1)} = \varphi_{i,j}^{(n)} + \frac{1}{4}\,R_{i,j}$$

Dans cette méthode, on est amené à poser :

$$\varphi_{i,j}^{(n+1)} = \varphi_{i,j}^{(n)} + \left(\frac{1}{4}\,\omega\right) R_{i,j}$$

où $\left(\frac{1}{4}\,\omega\right)$ est un terme positif compris entre $\frac{1}{4}$ et $\frac{1}{2}$ en pratique. Remarquons que la méthode précédente (Gauss-Seidel) revient à prendre $\omega = 1$.

En posant $a = \frac{1}{4}\,\omega$, il a été montré que la valeur optimale de a est la plus petite racine de l'équation :

$$a^2 z^2 - 4\,a + 1 = 0$$

dans laquelle :

$$z = \cos \frac{\pi}{i_M} + \cos \frac{\pi}{j_M}$$

i_M = valeur extrême (maximale) de l'indice i;

j_M = valeur extrême (maximale) de l'indice j.

Exemple

Reprenons l'exercice précédent et résolvons-le avec cette nouvelle méthode. Nous avons :

$$i_M = j_M = 10$$

L'équation du second degré précédente a deux racines 0,73 et 0,38. Cette dernière sera notre coefficient $\frac{1}{4}\,\omega$.

Le programme, modifié en conséquence, conduit beaucoup plus rapidement à la solution.

```
 5 DIM P(10,10):DIM Q(10,10): DIM R(10,1
0):N=0
 10 REM------------------------
 20 REM CONDITIONS AUX LIMITES
 30 REM------------------------
 40 FOR I=0 TO 10
 50 P(I,0)=10
 55 Q(I,0)=10
 60 NEXT
 70 FOR I=0 TO 10
 80 P(I,10)=100
 90 Q(I,10)=100
 95 NEXT
 100 FOR J = 0 TO 10
 110 P(0,J)=50
 115 Q(0,J)=50
 120 NEXT
 130 FOR J=0 TO 10
```

```
140 P(10,J)=50
150 Q(10,J)=50
160 NEXT
190 B=0.38
200 REM------------------------
210 REM ETAT INITIAL QUELCONQUE
220 REM------------------------
230 FOR I=1 TO 9
240 FOR J=1 TO 9
250 Q(I,J)=50
260 NEXT
270 NEXT
300 REM--------
310 REM CALCULS
320 REM--------
330 FOR J=1 TO 9
340 FOR I=1 TO 9
350 P(I,J)=Q(I,J)+B*(P(I-1,J)+Q(I+1,J)+
P(I,J-1)+Q(I,J+1)-4*Q(I,J))
360 NEXT I
370 NEXT J
380 N=N+1
390 PRINTN
400 REM---------------------
410 REM EVALUATION DU RESIDU
420 REM---------------------
430 I0=INT(RND(1)*9+1)
440 J0=INT(RND(1)*9+1)
450 R=P(I0-1,J0)+Q(I0+1,J0)+P(I0,J0-1)+
Q(I0,J0+1)-4*Q(I0,J0)
460 RS=R/P(I0,J0)
470 PRINTRS
500 REM----------
510 REM ITERATION
520 REM----------
530 FOR I=1 TO 9
540 FOR J=1 TO 9
545 R(I,J)=P(I-1,J)+Q(I+1,J)+P(I,J-1)+Q
(I,J+1)-4*Q(I,J)
550 Q(I,J)=P(I,J)
560 NEXT J
570 NEXT I
```

```
580 IF N=1 THEN GOSUB 1100
585 IF INT (N/5)=N/5 THEN GOSUB 1100
590 GOTO 300
1100 FOR J=10 TO 0 STEP -1
1110 FOR I=0 TO 10
1120 LPRINT TAB(I*6);INT(R(I,J)*10)/10;
1130 NEXT I
1140 LPRINT
1150 NEXT J
1160 LPRINT:LPRINT"Passage numero ";N
1165 LPRINT:LPRINT:LPRINT:LPRINT
1170 RETURN
```

VERSION COULEUR

```
 5 DIM P(11,11):DIM Q(11,11): DIM R(11,1
1):N=0
 10 REM-----------------------
 20 REM CONDITIONS AUX LIMITES
 30 REM-----------------------
 40 FOR I=0 TO 11
 50 P(I,0)=10
 55 Q(I,0)=10
 60 NEXT
 70 FOR I=0 TO 11
 80 P(I,11)=100
 90 Q(I,11)=100
 95 NEXT
 100 FOR J = 0 TO 11
 110 P(0,J)=50
 115 Q(0,J)=50
 120 NEXT
 130 FOR J=0 TO 11
 140 P(11,J)=50
 150 Q(11,J)=50
 160 NEXT
 200 REM-----------------------
 210 REM ETAT INITIAL QUELCONQUE
 220 REM-----------------------
 230 FOR I=1 TO 10
 240 FOR J=1 TO 10
```

```
 250 Q(I,J)=50
 260 NEXT
 270 NEXT
 280 HIRES: PAPER 2
 290 POKE 26,96
 300 REM--------
 310 REM CALCULS
 320 REM--------
 330 FOR J=1 TO 10
 340 FOR I=1 TO 10
 350 P(I,J)=.25*(P(I-1,J)+Q(I+1,J)+P(I,J
-1)+Q(I,J+1))
 360 NEXT I
 370 NEXT J
 380 N=N+1
 400 REM--------------------
 410 REM EVALUATION DU RESIDU
 420 REM--------------------
 430 I0=INT(RND(1)*10+1)
 440 J0=INT(RND(1)*10+1)
 450 R=P(I0-1,J0)+Q(I0+1,J0)+P(I0,J0-1)+
Q(I0,J0+1)-4*Q(I0,J0)
 460 RS=R/P(I0,J0)
 500 REM----------
 510 REM ITERATION
 520 REM----------
 530 FOR I=1 TO 10
 540 FOR J=1 TO 10
 545 R(I,J)=P(I-1,J)+Q(I+1,J)+P(I,J-1)+Q
(I,J+1)-4*Q(I,J)
 550 Q(I,J)=P(I,J)
 560 NEXT J
 570 NEXT I
 580 IF N=1 THEN GOSUB 1100
 585 IF INT (N/5)=N/5 THEN GOSUB 1100
 590 GOTO 300
 1100 FOR I=0 TO 11
 1105 FOR K=0 TO 2
 1110 CURSET 12+I*18+K*6,10,3
 1115 FOR J=11 TO 0 STEP -1
 1120 W1=(P(I,J)-10)*7.99/90:W2=W1-INT(W
1)
```

```
 1125 W=16+INT(W1)
 1130 IF (K=1) AND (W2>.66) AND(W<23)THE
N W=W+1
 1135 IF (K=1) AND (W2<.33) AND(W>0) THE
N W=W-1
 1140 FILL 15,1,W
 1145 NEXT
 1150 NEXT
 1155 NEXT
 1160 CURSET 228,10,3
 1170 FILL 180,2,18
 1200 PRINT "PASSAGE NUMERO: ";N
 1210 RETURN
```

Fonction

50	100	100	100	100	100	100	100	100	100	50
50	68.9	76.1	78.8	79.8	80.2	80.2	80.2	80.2	80.2	50
50	49.9	49.9	49.8	49.7	49.6	49.4	49.4	49.3	49.2	50
50	49.9	49.8	49.6	49.4	49.2	49	48.9	48.8	48.7	50
50	49.8	49.6	49.2	48.8	48.5	48.3	48.1	48	47.9	50
50	49.6	49	48.3	47.7	47.3	47	46.8	46.6	46.6	50
50	49.1	47.8	46.6	45.7	45.1	44.7	44.5	44.4	44.4	50
50	47.8	45.3	43.4	42.1	41.5	41.1	40.9	40.8	40.8	50
50	44.2	39.8	37.3	36	35.4	35.1	35	35	34.9	50
50	34.7	29	26.8	25.9	25.6	25.5	25.5	25.4	25.4	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 1

50	100	100	100	100	100	100	100	100	100	50
50	73.4	81.9	85.3	86.8	87.6	87.6	86	82.6	73.8	50
50	61.8	68.8	72.5	74.4	75.3	76.3	74.2	70.4	62.8	50
50	55.3	59.3	61.7	63.1	63.8	64.8	65.8	61.3	56.9	50
50	51	52.2	52.9	53.2	53.3	54.2	55.3	56.8	52.9	50
50	48	46.5	45.5	44.8	44.3	45.2	46.3	48.3	51	50
50	46.2	42.9	40.4	38.6	37.3	37.9	39.1	41.6	45.3	50
50	43.9	39.2	36	33.9	32.5	33.1	34.6	37.9	43	50
50	39.7	33.3	29.6	27.5	26.3	26.7	28.5	32.1	38.9	50
50	31	23.9	20.9	19.5	18.8	18.9	20.2	23.2	30.5	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 5

50	100	100	100	100	100	100	100	100	100	50
50	73.9	82.7	86.4	87.9	88.4	88	86.5	82.8	73.9	50
50	62.9	70.7	74.9	76.9	77.6	77	75.1	70.9	63	50
50	57	62.2	65.6	67.3	68	67.5	65.9	62.7	57.3	50
50	53.1	55.7	57.8	58.9	59.4	59.2	58.3	56.5	53.6	50
50	50	50.2	50.7	51.4	51.5	51.7	51.6	51.2	50.7	50
50	47.1	45.1	43.9	43.7	44.2	44.3	45.1	46.2	47.9	50
50	43.8	39.5	37	35.9	36	37.1	38.2	40.7	44.6	50
50	39.1	32.7	29.3	27.8	27.5	28.4	30.6	33.7	39.8	50
50	30.5	23.2	20.3	19	18.8	19.3	20.8	24	30.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 10

50	100	100	100	100	100	100	100	100	100	50
50	73.9	82.8	86.5	88	88.5	88	86.5	82.8	74	50
50	63	71	75.2	77.2	77.8	77.2	75.2	71	63.1	50
50	57.3	62.7	66.1	67.8	68.4	67.9	66.1	62.8	57.4	50
50	53.6	56.6	58.6	59.7	60	59.7	58.6	56.6	53.7	50
50	50.7	51.4	51.9	52.3	52.4	52.3	52	51.5	50.8	50
50	47.9	46.4	45.5	45.1	44.9	45.1	45.6	46.5	48	50
50	44.7	40.9	38.7	37.5	37.2	37.6	38.7	41	44.7	50
50	39.8	33.9	30.7	29.3	28.9	29.3	30.8	33.9	39.9	50
50	30.9	23.9	21.2	20	19.7	20	21.2	24	30.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 15

50	100	100	100	100	100	100	100	100	100	50
50	74	82.8	86.5	88	88.5	88	86.5	82.8	74	50
50	63.1	71	75.2	77.3	77.9	77.3	75.2	71	63.1	50
50	57.4	62.8	66.1	67.9	68.4	67.9	66.1	62.8	57.4	50
50	53.7	56.6	58.6	59.7	60.1	59.7	58.6	56.6	53.7	50
50	50.8	51.5	52	52.3	52.4	52.3	52	51.5	50.8	50
50	48	46.5	45.6	45.2	45	45.2	45.6	46.5	48	50
50	44.7	41	38.8	37.7	37.3	37.7	38.8	41	44.7	50
50	39.9	33.9	30.8	29.4	29	29.4	30.8	33.9	39.9	50
50	30.9	24	21.2	20.1	19.8	20.1	21.2	24	30.9	50
50	10	10	10	10	10	10	10	10	10	50

Passage numero 20

Résidus

0	0	0	0	0	0	0	0	0	0	0
0	49.9	68.9	76	78.6	79.4	79.6	79.6	79.6	79.5	0
0	-.1	-.2	-.5	-.8	-1.1	-1.4	-1.6	-1.8	-1.9	0
0	-.2	-.5	-1	-1.5	-2.1	-2.5	-2.8	-3.1	-3.2	0
0	-.4	-1.1	-2.1	-3	-3.9	-4.5	-4.9	-5.2	-5.4	0
0	-.9	-2.5	-4.3	-5.9	-7.1	-7.9	-8.5	-8.8	-8.9	0
0	-2.2	-5.6	-8.8	-11.2	-12.8	-13.7	-14.3	-14.6	-14.7	0
0	-5.8	-12.4	-17.4	-20.6	-22.4	-23.4	-23.8	-24.1	-24.2	0
0	-15.2	-26.8	-33.4	-36.7	-38.3	-39	-39.4	-39.5	-39.5	0
0	-40	-55.2	-61	-63.2	-64.1	-64.4	-64.5	-64.5	-64.6	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 1

0	0	0	0	0	0	0	0	0	0	0
0	.6	1.1	1.4	1.5	1.7	1	-.1	.4	.1	0
0	1.6	2.5	3.1	3.5	3.8	5.3	-2.3	1.6	.1	0
0	2.4	3.8	4.8	5.5	5.9	8.1	8	-6.4	1.6	0
0	3.2	5.4	6.4	6.8	7	9.7	9.3	10	-5.2	0
0	.1	1.4	3.1	4.6	5.8	9.8	9.2	10.3	10.8	0
0	-1.1	-1.9	-2.5	-3	-3.3	.2	-1.5	-.6	-.3	0
0	-1.2	-2	-2.6	-3	-3.4	.1	-1.4	-.5	-.3	0
0	-1.1	-1.8	-2.3	-2.7	-3	-.3	-.8	-.5	-.3	0
0	-.7	-1.2	-1.5	-1.8	-1.9	-.7	0	-.5	-.2	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 5

0	0	0	0	0	0	0	0	0	0	0
0	.3	.1	.1	.2	.1	.1	.1	0	0	0
0	.4	.6	.3	.4	.3	.3	.2	.1	0	0
0	.6	1	.9	.5	.7	.5	.3	.2	0	0
0	.7	1.3	1.7	.7	.9	.7	.5	.3	.1	0
0	.7	1.4	1.9	2.4	.3	1.3	.6	.4	.1	0
0	.6	1.4	1.9	2.4	2.8	0	1	.5	.1	0
0	.5	1.1	1.6	2.1	2.5	2.8	0	.4	.2	0
0	.2	.8	1.2	1.5	1.8	2.1	2.3	.2	-.1	0
0	-.1	.2	.5	.7	1	1.1	1.3	1.4	-.1	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 10

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	.1	.1	0	0	0	0	0	0	0
0	0	.1	.1	.1	.1	0	0	0	0	0
0	0	.1	.1	.1	.1	.1	0	0	0	0
0	.3	.1	.2	.1	.1	.1	0	0	0	0
0	.2	.4	0	.1	.1	.1	0	0	0	0
0	.1	.2	.3	-.3	.1	.1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 15

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	-.1	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	-.1	0	0	0	0	0	0	0	0	0
0	-.1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 20

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Passage numero 25

VI
UTILITAIRES

17. TRI DE FICHIERS

Il s'agit d'un processus important dans la manipulation de fichiers.

Soit un ensemble de N nombres dans un ordre quelconque. Supposons-les placés dans une matrice A (N). Le but de l'algorithme est de ranger dans l'ordre croissant (ou décroissant) cet ensemble de nombres. Différentes méthodes existent; certaines sont beaucoup plus rapides que d'autres; le critère de sélection est naturellement le nombre N. Si ce nombre est grand, il faut utiliser une méthode sophistiquée et rapide; s'il est petit, une méthode plus simple convient.

Le tri de mots est effectué de la même manière. En effet, pour un ordinateur, les lettres ont un *ordre*, identique à l'ordre alphabétique, défini par leur code ASCII. Ainsi :

ARBALETE < ARBRE < FORET < ZOO

Cependant, des problèmes apparaissent si minuscules et majuscules sont mélangées. On a :

Majuscule < minuscule

Ainsi :

... W < X < Y < Z < *a* < *b* < *c*...

Cela étant dit, les algorithmes restent les mêmes, dans la mesure où A (I) est remplacé par A$ (I).

L'opération la plus souvent effectuée dans ce genre de problèmes est l'échange de deux données. Certains micro-ordinateurs disposent pour cela de l'instruction SWAP.

On peut ranger les différentes méthodes de tri dans trois catégories principales :

a. Tris par échange (ou par inversion)

Le principe est de comparer et d'inverser (éventuellement) deux éléments de la liste suivant une règle fixe.
Appartiennent à cette catégorie :

— Ripple
— Bubble
— Shaker
— Quick

b. Tris par insertion

Chaque élément est inséré à la bonne place. Citons :

— Insertion directe
— Shell
— Shell-Metzner

c. Tris par sélection

On sélectionne et on classe l'élément le plus grand successivement dans plusieurs sous-listes, en arbres. Ce sont :

— Sélection
— Heap sort

Nous verrons en détail ces différentes méthodes, en donnant des exemples commentés. Puis nous verrons l'intérêt d'avoir des listes classées : l'accès à un élément particulier en est grandement facilité. Nous aborderons enfin l'accès plus général en donnant simplement les algorithmes dans leurs grandes lignes. Il est en effet ici impossible de donner des programmes sans faire appel aux propriétés particulières du DOS de la machine et des types (séquentiels ou à accès direct) de fichiers qu'il peut traiter.

Méthodes de tri

A. Tris par échange

Ripple sort

Il s'agit de faire (N — 1) *passes* sur l'ensemble de la matrice, en effectuant la comparaison systématique de deux éléments consécutifs, en inversant la place de ceux-ci s'ils ne sont pas dans l'ordre. Le nombre de comparaisons effectuées est de l'ordre de N^2 (O (N^2)).

En particulier, si A (N) est déjà rangée, l'algorithme ne le « sait » pas et effectue donc le même nombre de comparaisons (ce qui est un inconvénient). Cependant, l'utilisation d'un test (lignes 130, 170, 200, 210) permet d'éviter ce phénomène puisqu'il est ainsi possible de savoir si, au cours d'une passe, il y a eu permutation à un moment quelconque. Dans le cas contraire, c'est que le tri est terminé.

Programme

```
9 REM ------------------------------------------
10 REM CONSTRUCTION DE LA MATRICE
11 REM ------------------------------------------
20 INPUT "NOMBRE N= ";N
30 DIM A(N)
40 FOR I=1 TO N
50 A(I)=INT(RND(1)*1000)
60 NEXT
99 REM ----------------------
100 REM RIPPLE SORT
101 REM ----------------------
110 I=N
120 I=I-1
130 TEST=0
140 FOR J=1 TO I
150 IF A(J)=<A(J+1) THEN 180
160 U=A(J):A(J)=A(J+1):A(J+1)=U
170 TEST=1
180 NEXT
200 IF TEST=0 THEN STOP
210 IF TEST=1 THEN 120
230 FOR I=1 TO N:PRINTA(I);:NEXT
```

Exemple de fonctionnement

Soit à classer une matrice de 15 éléments, dans l'ordre croissant d'abord, dans l'ordre décroissant ensuite. Le résultat obtenu est le suivant :

```
676  351  348  713  540  406  321  15   388  22   127  290  526  409  494

351  348  676  540  406  321  15   388  22   127  290  526  409  494  713
348  351  540  406  321  15   388  22   127  290  526  409  494  676  713
348  351  406  321  15   388  22   127  290  526  409  494  540  676  713
348  351  321  15   388  22   127  290  406  409  494  526  540  676  713
348  321  15   351  22   127  290  388  406  409  494  526  540  676  713
321  15   348  22   127  290  351  388  406  409  494  526  540  676  713
15   321  22   127  290  348  351  388  406  409  494  526  540  676  713
15   22   127  290  321  348  351  388  406  409  494  526  540  676  713
```

En remplaçant (ligne 150) le signe de comparaison ⩽ par ⩾, on réalise un tri inverse :

```
676  351  348  713  540  406  321  15   388  22   127  290  526  409  494

676  351  713  540  406  348  321  388  22   127  290  526  409  494  15
676  713  540  406  351  348  388  321  127  290  526  409  494  22   15
713  676  540  406  351  388  348  321  290  526  409  494  127  22   15
713  676  540  406  388  351  348  321  526  409  494  290  127  22   15
713  676  540  406  388  351  348  526  409  494  321  290  127  22   15
713  676  540  406  388  351  526  409  494  348  321  290  127  22   15
713  676  540  406  388  526  409  494  351  348  321  290  127  22   15
713  676  540  406  526  409  494  388  351  348  321  290  127  22   15
713  676  540  526  409  494  406  388  351  348  321  290  127  22   15
713  676  540  526  494  409  406  388  351  348  321  290  127  22   15
```

Attention

Les exemples sont donnés à titre indicatif, et permettent d'interpréter la manière dont le classement évolue en fonction du temps. Ils ne sont pas significatifs de la vitesse à laquelle évolue le tri.

Commentaires

Étudions la manière dont fonctionne le programme. Pour N donné, on effectue N — 1 comparaisons sur l'ensemble des éléments à trier. Au premier passage, chaque élément est comparé au suivant. Il y a échange si l'ordre n'est pas le bon. L'algorithme enrichit donc la tête de liste d'un seul élément (le plus grand parmi les éléments non triés) à chaque passage. On conçoit donc que le nombre (maximal) de comparaisons à effectuer est N*(N — 1).

Cependant, étant sûr que la partie triée augmente d'un élément à chaque passage, on peut (ligne 210) réduire simultanément d'un élément la nouvelle liste à trier.

Nous ajoutons un petit programme qui trie une liste contenant des noms et des chiffres, une fois par ordre alphabétique, et ensuite par ordre numérique.

Exemple

				1			2
DURAND	33			ABEL	35	31	ISOBAR
DUPONT	32			DUPONT	32	32	DUPONT
ABEL	35	}	⟶	DURAND	33	33	DURAND
ISOBAR	31			ISOBAR	31	35	ABEL
LION	41			LION	41	41	LION

Programme

```
 5 REM RIPPLE PLUSIEURS CRITERES
 10 INPUT "NOMBRE N= ";N
 20 DIM A$(N),A(N)
 30 FOR I=1 TO N
 40 PRINT "NOM A$(";I;") = ";:INPUT A$(I
)
 50 PRINT "VALEUR A(";I;") = ";:INPUT A(
I)
 60 NEXT
 80 INPUT "TRI A$, A, FIN (A/B/F) ";Q$
 90 IF Q$<>"A" AND Q$<>"B" THEN END
 100 REM TRI RIPPLE
 120 P=N
 130 P1=P: FOR I=1 TO P1-1
 135 U$=A$(I+1):U=A(I+1)
 140 IF Q$="A" AND U$>=A$(I) THEN 160
 145 IF Q$="B" AND U>=A(I)  THEN 160
 150 A$(I+1)=A$(I):A$(I)=U$
 155 A(I+1)=A(I): A(I)=U:P=I
 160 NEXT
 170 IF P1=P THEN 200
 180 GOTO 130
```

```
 200 FOR I=1 TO N
 205 IF Q$="B" THEN PRINT A(I), A$(I):GO
TO220
 210 PRINTA$(I),A(I)
 220 NEXT
 230 GOTO 80
```

Le paramètre P1 indique s'il y a eu inversion, sinon la liste est en ordre, le tri est terminé. Le paramètre P marque la position à partir de laquelle les éléments sont classés.

Quand on trie une liste de chaînes A$(I) à l'aide d'un programme en BASIC il faut prendre en considération — surtout si le nombre N est très grand — le fait qu'il y a de temps en temps un ramassage de déchets *(Garbage collection)* qui prend beaucoup de temps. Il vaut donc mieux introduire un vecteur clé B(I) et ne pas inverser les variables chaînes.

Voici le petit programme qui trie les indices au lieu des chaînes :

```
 5 REM RIPPLE PLUSIEURS CRITERES-ACCES P
AR CLE
 10 INPUT "NOMBRE N= ";N
 20 DIM A$(N),A(N),B(N)
 30 FOR I=1 TO N
 40 PRINT "NOM A$(";I;") = ";:INPUT A$(I
)
 50 PRINT "VALEUR A(";I;") = ";:INPUT A(
I):B(I)=I
 60 NEXT
 80 INPUT "TRI A$, A, FIN (A/B/F) ";Q$
 90 IF Q$<>"A" AND Q$<>"B" THEN END
 100 REM TRI RIPPLE
 120 P=N
 130 P1=P: FOR I=1 TO P-1
 135 V=B(I+1):U$=A$(V):U=A(I+1)
 140 IF Q$="A" AND U$>=A$(B(I))THEN 160
 145 IF Q$="B" AND U>=A(I)  THEN 160
 150 B(I+1)=B(I):B(I)=V
 155 A(I+1)=A(I): A(I)=U:P=I
 160 NEXT
 170 IF P1=P THEN 200
```

```
180 GOTO 130
200 FOR I=1 TO N
205 IF Q$="B" THEN PRINT A(I), A$(B(I))
:GOTO220
210 PRINTA$(B(I)),A(I)
220 NEXT
230 GOTO 80
```

Remarque

Un tri Ripple est très efficace sur une liste déjà ordonnée au départ.

Tri Shaker

On peut essayer d'améliorer le procédé du Ripple en commençant les comparaisons une fois par la gauche et immédiatement après par la droite. Le succès dudit Shaker sort n'est pourtant pas évident.

Voici le programme :

Les paramètres D et F limitent la sous-liste qui reste à trier. A noter une petite acrobatie de BASIC (Microsoft) autorisée par la méthode de stockage des variables D et F (valeurs initiales et finales des boucles) d'une boucle FOR ... NEXT. 160 ... F = I ne modifie pas la valeur finale de la boucle en train. Vérifiez-la sur votre machine.

```
10 REM SHAKER RIPPLE
20 INPUT "NOMBRE N= ";N
30 DIM A(N)
40 FOR I=1 TO N
50 A(I) = INT(RND(+1)*1000)
60 NEXT
100 REM TRI SKAKER
110 D=1:F=N
120 TEST=0
130 FOR I=D TO F-1
```

```
 150 IFA(I)<=A(I+1) THEN 170
 160 U= A(I):A(I)=A(I+1):A(I+1)=U:TEST=1
:F=I
 170 NEXT
 180 IF TEST =0 THEN 300
 185 TEST=0
 190 FOR I=F TO D+1 STEP -1
 210 IFA(I)>=A(I-1) THEN 230
 220 U=A(I):A(I)=A(I-1):A(I-1)=U:TEST=1:
D=I
 230 NEXT
 240 IF TEST=0 THEN 300
 250 GOTO120
 300 FOR I=1 TO N
 310 PRINT A(I);
 320 NEXT
 330 STOP
```

Tri Shaker : Exemple

```
      Tri SHAKER  exemple                                     Paramètres D ou  F
676  351  348  713  540  406  321  15  388  22  127  290  526  409  494         ↘

351  348  676  540  406  321  15  388  22  127  290  526  409  494  713    14
15  351  348  676  540  406  321  22  388  127  290  409  526  494  713    2
15  348  351  540  406  321  22  388  127  290  409  526  494  676  713    13
15  22  348  351  540  406  321  127  388  290  409  494  526  676  713    3
15  22  348  351  406  321  127  388  290  409  494  526  540  676  713    12
15  22  127  348  351  406  321  290  388  409  494  526  540  676  713    4
15  22  127  348  351  321  290  388  406  409  494  526  540  676  713    8
15  22  127  290  348  351  321  388  406  409  494  526  540  676  713    5
15  22  127  290  348  321  351  388  406  409  494  526  540  676  713    6
15  22  127  290  321  348  351  388  406  409  494  526  540  676  713    6
```

Commentaire

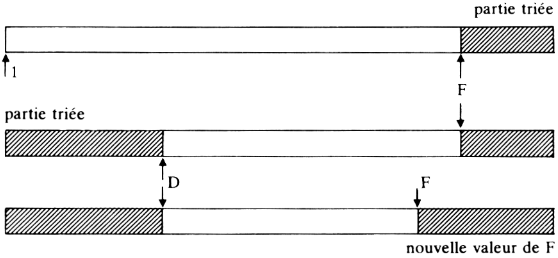
Cette méthode de tri porte un nom qui indique bien la manière dont elle opère. Le premier passage est classique, du type Ripple. Le plus grand élément est amené en début de liste. Le second passage est un Ripple inverse. Partant du début de la liste, on ramène à la fin le plus petit.

Entre-temps, la liste en question a été réduite à la partie non triée, et cela se produira à chaque passage :

Premier passage ⟶
Ripple direct

Deuxième passage ⟶
Ripple inverse

Troisième passage ⟶
Ripple direct



et ainsi de suite...

On pourrait dire que F et D jouent le rôle de mémoires, conservant pour la prochaine étape du tri l'indice du tableau pour lequel le dernier échange a eu lieu.

Bubble sort (tri à bulle)

Le principe est le suivant :

- Dans la liste des N nombres, on recherche le plus petit. Celui-ci est alors placé en tête (premier élément).
- Dans la liste des (N — 1) nombres restants, à partir du second, on recherche le plus petit. Il est placé en tête de cette nouvelle liste et devient en fait le deuxième élément de la liste triée.
- Dans la liste des (N — 2) nombres restants, ... etc.

On conçoit que les éléments remontent, dans l'ordre du tri, comme des bulles dans un tube à essai, d'où le nom de la méthode.

Programme

```
9 REM ------------------------------
10 REM CONSTRUCTION DE LA MATRICE
11 REM ------------------------------
20 INPUT "NOMBRE N= ";N
30 DIM A(N)
40 FOR I=1 TO N
```

```
50 A( I )=INT( RND( 1 )*1000 )
60 NEXT
99 REM  -----------
100 REM BUBBLE SORT
101 REM -----------
120 FOR I=1 TO N-1
130 FOR J=I+1 TO N
140 IF A( I )=<A( J ) THEN 160
150 U=A( I ):A( I )=A( J ):A( J )=U
160 NEXT J
170 NEXT I
180 STOP
200 FOR I=1 TO N:PRINTA( I );:NEXT
```

Exemple de fonctionnement

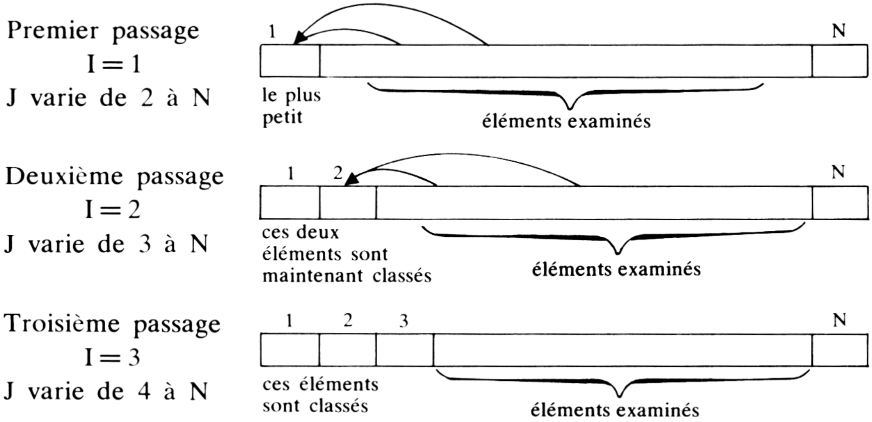
En reprenant la matrice précédente, on réalise ce programme. On peut constater que les éléments remontent dans l'ordre vers le haut : à la I[ième] passe, les I premiers éléments sont déjà rangés.

```
676  351  348  713  540  406  321  15   388  22   127  290  526  409  494

15   676  351  713  540  406  348  321  388  22   127  290  526  409  494
15   22   676  713  540  406  351  348  388  321  127  290  526  409  494
15   22   127  713  676  540  406  351  388  348  321  290  526  409  494
15   22   127  290  713  676  540  406  388  351  348  321  526  409  494
15   22   127  290  321  713  676  540  406  388  351  348  526  409  494
15   22   127  290  321  348  713  676  540  406  388  351  526  409  494
15   22   127  290  321  348  351  713  676  540  406  388  526  409  494
15   22   127  290  321  348  351  388  713  676  540  406  526  409  494
15   22   127  290  321  348  351  388  406  713  676  540  526  409  494
15   22   127  290  321  348  351  388  406  409  713  676  540  526  494
15   22   127  290  321  348  351  388  406  409  494  713  676  540  526
15   22   127  290  321  348  351  388  406  409  494  526  713  676  540
15   22   127  290  321  348  351  388  406  409  494  526  540  713  676
15   22   127  290  321  348  351  388  406  409  494  526  540  676  713
```

Commentaire

Le premier élément est comparé à tous les autres (et, le cas échéant, échangé avec ceux-ci). Le plus petit a donc obligatoirement pris sa place à la fin du premier passage. Le second élément est ensuite comparé à tous les autres et, par le même procédé, remplacé par le plus petit de la liste restante. Ce phénomène se poursuit jusqu'au classement final.



Remarque

On pourrait se demander quelle différence existe entre un tri Ripple et un tri Bubble. Celle-ci est subtile. Bubble compare successivement et systématiquement chaque élément de la liste à tous les autres. Ripple fait de même, mais sur deux éléments consécutifs. Cela fait qu'il se constitue à chaque passage des micro-listes triées, dans la mesure où un élément temporairement *grand* (devant ceux auxquels il a été comparé) remonte la liste jusqu'au moment où il en rencontre un plus grand que lui. Pour poursuivre l'analogie hydraulique, il y a remontée des petites « bulles »; celles-ci s'arrêtent dès qu'elles en ont rencontré une plus grosse.

La méthode Bubble se justifie lorsque le tri n'a pas besoin d'être complet; on se contente, par exemple, de ne connaître que les trois « premiers » d'une liste.

Quick sort (tri rapide)

Il s'agit d'une méthode sophistiquée, très adaptée au traitement des fichiers importants. Dans ce cas, le temps de tri peut être nettement inférieur à celui que nécessitent les autres méthodes.

Programme

```
 0 GOTO 10
 1 REM ****************************
 2 REM * INITIALISATION TABLE NOMBRES *
 3 REM ****************************
 4 REM
 7 X=A(U):A(U)=A(S):A(S)=X
 8 RETURN
 9 REM
 10 INPUT "NOMBRE N= ";N
 11 N1=INT(3*LN(N))
 12 DIM A(N)
 13 INPUT "LONGEUR MICRO-LISTES 4=<LP=<1
0 :";LP
 15 INPUT "TRI FINAL RIPPLE / INSERTION
(R/I) :"; TF$
 20 FOR I=1 TO N
 30 A(I) = INT(RND(1)*1000)
 40 NEXT
 45 PRINT "DEBUT TRI"
 50 REM
 51 REM ************************
 52 REM * INITIALISATION VARIABLES *
 53 REM ************************
 54 REM
 60 DIM PLE(1,N1)
 70 PP=0:BAS=1:HAUT=N
 80 PVOT=0:MIL=0
 90 REM
 99 REM  *****************
 100 REM * CHOIX DU PIVOT *
 101 REM *****************
 102 REM
 110 MIL=INT(BAS+HAUT)/2
 120 IF A(BAS)<A(MIL) THEN U=BAS:S=MIL:G
OSUB7:I=0
 130 IF A(HAUT)<A(BAS) THEN U=BAS:S=HAUT
:GOSUB7:I=1
 140 IF I=1 AND A(BAS)<A(MIL) THEN U=MIL
:S=BAS:GOSUB 7
 150 REM
```

```
151 REM ****************************
152 REM * CLASSEMENT DES ELEMENTS *
153 REM *      AUTOUR DU PIVOT      *
154 REM ****************************
155 REM
160 PVOT=A(BAS)
170 PB=BAS:PH=HAUT+1
180 REPEAT
190 PB=PB+1
200 UNTIL A(PB)>=PVOT
210 REPEAT
220 PH=PH-1
230 UNTIL A(PH)<=PVOT
240 IF PB>=PH THEN 270
250 U=PB:S=PH:GOSUB7
260 GOTO 180
270 U=BAS:S=PH:GOSUB7
280 REM
281 REM ****************************
282 REM * ORGANISATION DE LA PILE *
283 REM ****************************
284 REM
290 IF (PH-BAS)>(HAUT-PH) THEN 400
298 REM
300 IF (PH-BAS)<=LP THEN 360
310 PP=PP+1
320 PLE(0,PP)=PH+1
330 PLE(1,PP)=HAUT
340 HAUT=PH-1
350 GOTO110
360 IF (HAUT-PH)<=LP THEN 500
370 BAS=PH+1
380 GOTO110
395 REM
400 IF (HAUT-PH)<=LP THEN 460
410 PP=PP+1
420 PLE(0,PP)=BAS
430 PLE(1,PP)=PH-1
440 BAS=PH+1
450 GOTO 110
460 IF (PH-BAS)<=LP THEN 500
470 HAUT=PH-1
480 GOTO 110
```

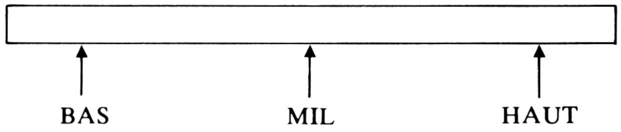
```
490 REM
491 REM ***********
492 REM * DEPILER *
493 REM ***********
494 REM
500 IF PP=0 THEN 580
510 BAS=PLE(0,PP)
520 HAUT=PLE(1,PP)
530 PP=PP-1
540 GOTO 110
580 IF TF$="I" THEN 700
587 REM
589 REM ********************
590 REM * TRI FINAL RIPPLE *
591 REM ********************
592 REM
600 I=N
610 PRINT"DEBUT RIPPLE"
620 REPEAT
630 TEST=0
640 FOR J=1 TO I-1
650 IFA(J)=<A(J+1) THEN 680
660 X=A(J):A(J)=A(J+1):A(J+1)=X
670 TEST=1
680 NEXT
685 I=I-1
690 UNTIL TEST=0
691 GOTO 800
695 REM
696 REM ***********************
697 REM * TRI FINAL INSERTION *
698 REM ***********************
699 REM
700 PRINT"DEBUT INSERTION"
710 FOR J=2 TO N
720 IF A(J)>=A(J-1) THEN 770
730 U=A(J):G=J-1
740 REPEAT
750 A(G+1)=A(G): A(G)=U:G=G-1
760 UNTIL G=0 OR U>=A(G)
770 NEXT
800 FOR I=1 TO N:PRINT A(I);:NEXT
```

Commentaires

Soit une liste de 256 éléments à trier, rangés dans A(256).

- Le programme débute ainsi :
  - — Définition d'une table pile PLE(1,16) (N=256, N1=INT(3*LN(N))=16).
  - — Définition d'un pointeur de pile PP (=0).
  - — BAS et HAUT : pointeurs d'extrémité de partition à trier. (Au départ : BAS=0 et HAUT=N=256.)
  - — PVOT : détermine la valeur du pivot (=0).
  - — MIL : pointe le milieu de la table à trier (=128).
  - — N : nombre d'éléments à classer (=256).
  - — PB, PH : pointeurs bas et haut.

- Le tri commence ligne 110 :



- — A(BAS) et A(MIL) sont comparés. Le plus grand des deux est placé en BAS; I=0.
- — A(HAUT) et A(BAS) sont comparés, le plus grand des deux est placé en HAUT; I=1.
- — A(BAS) et A(MIL) sont comparés si I=1; en effet, dans ce cas, il n'est pas certain que A(BAS)>A(MIL).

Prenons un exemple.

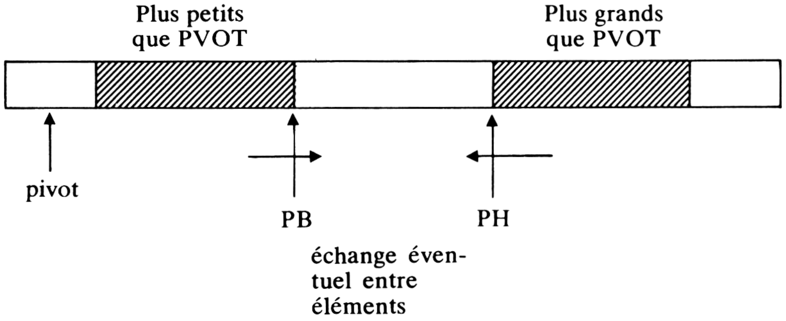
• Au départ :	9	14	6	I=0
• Après exécution de la ligne 120 :	14	9	6	I=0
• Après exécution de la ligne 130 :	6	9	14	I=1
• Après exécution de la ligne 140 :	9	6	14	

Le principe de la suite des opérations est de trouver un *pivot*, élément de la table ayant une valeur moyenne telle qu'il y ait par rapport à lui autant d'éléments plus petits que d'éléments plus grands. Dans notre exemple, le pivot est choisi parmi le premier, le dernier et le médian de la liste. Il est placé en tête de liste.

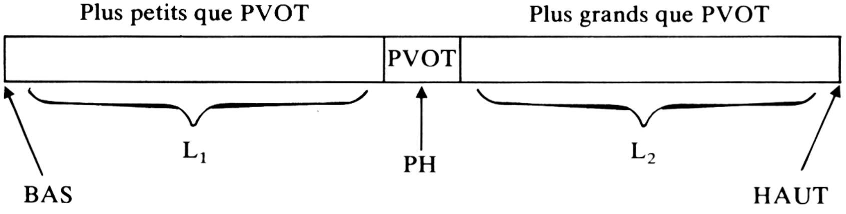
On a donc :

PVOT = A (BAS)

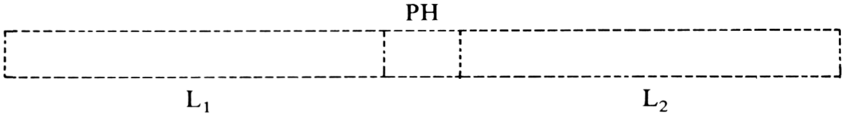
- Tous les éléments sont classés par référence au pivot. Partant du bas de la liste, on remonte celle-ci jusqu'à trouver un élément plus grand que le pivot, d'indice PB. De même, partant du haut de la liste, on descend celle-ci jusqu'à trouver un élément plus petit que le pivot :



Si PB ⩾ PH, toute la liste est répartie. Le programme peut se poursuivre. Sinon, les deux éléments A (PB) [plus grand que PIVOT] et A (PH) [plus petit que PIVOT] sont interchangés. Et le même cycle recommence. Cela continue jusqu'au moment où PB ⩾ PH, c'est-à-dire quand tous les éléments plus petits que PVOT sont d'un côté de la liste et que tous les éléments plus grands que PVOT sont de l'autre côté. A ce moment (ligne 270), le PVOT est placé en *séparateur* entre les deux demi-listes.



Après chaque classement autour du pivot, quatre tests sont effectués. Au premier passage, la liste concernée est la liste primitive, scindée en deux autour du pivot.



290 * $L_1 > L_2 \longrightarrow 400$

$L_1 < LP \longrightarrow 360$

Donc $L_1 \leqslant L_2 \longrightarrow$ mémoriser la position de L_2 et retrier L_1.

360 * $L_2 < LP \longrightarrow 500$

Sinon retrier L_2.

400 * $L_2 < LP \longrightarrow 460$

Sinon mémoriser L_1 et retrier L_2.

460 * $L_1 < LP \longrightarrow 500$

Sinon retrier L_1.

500 : Les divers tronçonnages de la liste primitive ont conduit à une liste de longueur inférieure à LP ($4 \leqslant LP \leqslant 10$). On vient maintenant récupérer les listes mises de côté à chaque étape.

- Si la liste L_1 est plus longue que la liste L_2, on renvoie en 400.
- Si la liste L_1 contient moins de LP éléments, on renvoie en 360.
- Le cas traité est donc :

longueur de $L_1 <$ longueur de L_2

c'est-à-dire que la demi-liste la plus courte est traitée. On a alors :

$PP = PP + 1 = 1$

$PLE(0, PP) = PLE(0, 1) = PH + 1$

(pointe le début de L_2)

$PLE(1, PP) = PLE(1, 1) = HAUT$

(pointe la fin de L_2)

Le programme renvoie alors en 110, pour effectuer un tri sur L_1, comme il avait été effectué au départ sur $L_1 + L_2$ (choix d'un pivot dans L_1, répartitions des éléments de L_1 autour de ce pivot).

Cela se poursuit jusqu'au moment où la dernière répartition conduit à des micro-listes contenant moins de LP éléments.

- En ligne 400, pour laquelle on a déjà $L_1 > L_2$, la longueur de L_2 est testée : si elle est inférieure à LP, il y a renvoi en 460. Sinon :

$$PP = PP + 1 = 1$$

$$PLE(0, PP) = PLE(0, 1) = BAS$$

$$PLE(1, PP) = PLE(1, 1) = PH - 1$$

Il y a mémorisation des coordonnées de L_1 et retour en 110 pour traiter L_2 (choix d'un pivot dans L_2, répartition des éléments de L_2 autour de ce pivot).

- En ligne 460, si les micro-listes de L_1 sont d'une longueur inférieure à LP, il y a sortie de ce type de tri. Sinon, on recommence en 110.
- En 500 : test de PP. Si $PP = 0$, c'est que toutes les micro-listes ont une longueur inférieure à LP; le tri est presque terminé. Sinon, il faut chercher quelle est la liste de longueur supérieure à LP et effectuer un nouveau traitement.
- En ligne 580 nous disposons maintenant d'un ensemble de listes de LP éléments prétriées et prérangées. Un tri final Ripple ou Insertion effectue le dernier classement. Il est très efficace à ce moment-là.

Nous préférons pour les listes importantes selon la proposition de D. Knuth $LP = 9$ et un tri final Insertion. Le choix est légitimé par une étude mathématique.

Prenons un exemple :

Soit la liste de vingt nombres donnée ci-après. Le pivot est $A(1) = 13$. (C'est une coïncidence.) Il est donc déjà placé en tête de liste et l'on n'a

pas à effectuer ce transfert. Suivons le déroulement du programme pas à pas :

1. * 84($>$13) est échangé avec 11 ($<$13)
   * 21 est échangé avec 2.

   Le passage est terminé; 2 est maintenant échangé avec le pivot 13. Nous avons maintenant affaire à deux listes; l'une L1, à gauche du pivot, de longueur inférieure à 4 est estimée *prétriée.*
   Elle ne sera plus touchée.

2. On s'occupe alors de L2, s'étendant de A(5) à A(20). Comme précédemment, le pivot est choisi dans cette liste :

$$A\left(\frac{(20+5)}{2}\right)=A(12)=82,\ A(5)=85,\ A(20)=89$$

   d'où le pivot 85, placé en tête de L2.
   Alors :

3. * 88 est échangé avec 84
   * Le pivot 85 est échangé avec la position pointée par PH, soit avec 17.

4. Nous avons maintenant deux nouvelles listes, que nous noterons L1 et L2, situées de part et d'autre du pivot.
   La plus longue, L1, est mémorisée, c'est-à-dire que ses indices de début et de fin sont placés dans une *pile* (tableau), qui permettra au programme par la suite de retrouver les tronçons de liste non triés pour pouvoir les traiter.

5. Le pivot est maintenant choisi dans L2, entre 89, 88 et 99. Il vaut donc 89 et il est placé en tête de liste. 88 et 94 sont alors interchangés, puis 89 avec 88. Nous aboutissons à deux nouvelles listes, de part et d'autre du pivot, mais de longueurs inférieures à 4. Elles sont estimées *prétriées.*

6. La liste A (5)-A (14) est alors dépilée, le pivot est 24 (choix entre 24, 14, 84).

7. 42 est échangé avec 21, puis 17 est échangé avec 24. Nous aboutissons à deux listes, dont l'une (L1 de A (5) à A (9)) est de longueur inférieure à 4. Elle est abandonnée.
   L2 (de A (10) à A (14)) est triée : pivot 82. Un seul échange a alors lieu : 82 <−> 73.

8. Le tri Quick est fini. Un tri Ripple conclut.

(0)

```
 13  94  84  21  82  18  42  14  97  2   70  89  73  88  17  11  4   99  24  85
 13  94  84  21  82  18  42  14  97  2   70  89  73  88  17  11  4   99  24  85
HAUT= 20         BAS= 1  PB= 0   PH= 0
```

(1)

```
                              PIVOT= 13
 13  4   84  21  82  18  42  14  97  2   70  89  73  88  17  11  94  99  24  85
 13  4   11  21  82  18  42  14  97  2   70  89  73  88  17  84  94  99  24  85
 13  4   11  2   82  18  42  14  97  21  70  89  73  88  17  84  94  99  24  85
 2   4   11  13  82  18  42  14  97  21  70  89  73  88  17  84  94  99  24  85
FIN DE TRI
```

(2)

```
L2 est triee: 5 - 20
 2   4   11  13  85  18  42  14  97  21  70  82  73  88  17  84  94  99  24  89
HAUT= 20         BAS= 5  PB= 5   PH= 4
```

(3)

```
                              PIVOT= 85
 2   4   11  13  85  18  42  14  24  21  70  82  73  88  17  84  94  99  97  89
 2   4   11  13  85  18  42  14  24  21  70  82  73  84  17  88  94  99  97  89
 2   4   11  13  17  18  42  14  24  21  70  82  73  84  85  88  94  99  97  89
FIN DE TRI
```

(4)

```
L1 est memorisee: 5 - 14
L2 est triee: 16 - 20
 2   4   11  13  17  18  42  14  24  21  70  82  73  84  85  89  94  88  97  99
HAUT= 20         BAS= 16          PB= 16  PH= 15
```

(5)

```
                              PIVOT= 89
 2   4   11  13  17  18  42  14  24  21  70  82  73  84  85  89  88  94  97  99
 2   4   11  13  17  18  42  14  24  21  70  82  73  84  85  88  89  94  97  99
FIN DE TRI
```

(6)

```
La liste : 5 - 14  est depilee.
 2   4   11  13  24  18  42  14  17  21  70  82  73  84  85  88  89  94  97  99
HAUT= 14         BAS= 5  PB= 18  PH= 17
```

(7)

```
                              PIVOT= 24
 2   4   11  13  24  18  21  14  17  42  70  82  73  84  85  88  89  94  97  99
 2   4   11  13  17  18  21  14  24  42  70  82  73  84  85  88  89  94  97  99
FIN DE TRI
```

```
(8)
L2 est triee: 10 - 14
 2   4   11  13  17  18  21  14  24  82  70  42  73  84  85  88  89  94  97  99
HAUT= 14          BAS= 10          PB= 10  PH= 9

(9)
                            PIVOT= 82
 2   4   11  13  17  18  21  14  24  73  70  42  82  84  85  88  89  94  97  99
FIN DE TRI

TRI TERMINE
```

B. Tris par insertion

Insertion directe

Soit une liste de N éléments à trier. On inverse les éléments A (1) et A (2), si nécessaire, et on range après les éléments A (3), A (4), ..., A (N) en commençant par A (3) et en comparant aux précédents. Voici le programme :

```
10 REM TRI PAR INSERTION
20 INPUT "NOMBRE N= ";N
30 DIM A(N)
40 FOR I=1 TO N
50 A(I) = INT (RND( 1)*1000)
55 PRINT A(I);
60 NEXT
100 REM TRI
110 FOR  J=2 TO N
120 IFA(J)>=A(J-1)THEN 170
130 U=A(J):  G=J-1
140 REPEAT
150 A(G+1)=A(G):  A(G)=U:G=G-1
160 UNTIL G=0 OR U>=A(G)
170 NEXT
180 PRINT
200 FOR I=1 TO N
210 PRINT A(I);
220 NEXT
```

Tri par insertion : Exemple

```
676  351  348  713  540  406  321  15   388  22   127  290  526  409  494

351  676  348  713  540  406  321  15   388  22   127  290  526  409  494
348  351  676  713  540  406  321  15   388  22   127  290  526  409  494
348  351  540  676  713  406  321  15   388  22   127  290  526  409  494
348  351  406  540  676  713  321  15   388  22   127  290  526  409  494
321  348  351  406  540  676  713  15   388  22   127  290  526  409  494
15  321  348  351  406  540  676  713  388  22   127  290  526  409  494
15  321  348  351  388  406  540  676  713  22   127  290  526  409  494
15  22  321  348  351  388  406  540  676  713  127  290  526  409  494
15  22  127  321  348  351  388  406  540  676  713  290  526  409  494
15  22  127  290  321  348  351  388  406  540  676  713  526  409  494
15  22  127  290  321  348  351  388  406  526  540  676  713  409  494
15  22  127  290  321  348  351  388  406  409  526  540  676  713  494
15  22  127  290  321  348  351  388  406  409  494  526  540  676  713
```

Tri Shell

Cette méthode rappelle le tri à bulles. La liste est scindée en deux parties (à peu près égales). Les premiers éléments sont comparés aux derniers. Puis la liste est scindée en quatre. Même opération, et ainsi de suite. Le *pas* de travail est ainsi de plus en plus petit, ce qui conduit à une diminution dynamique du nombre de comparaisons.

Programme

```
10 REM TRI SHELL
20 INPUT" NOMBRE N= ";N
30 DIM A(N)
40 FOR I =1 TO N
50 A(I) = INT(RND(1)*1000)
55 PRINT A(I);
60 NEXT
65 PRINT
99 REM ----------------
100 REM SHELL SORT
101 REM ----------------
110 D= INT(N/2)
130 TEST=0
140 FORI=1 TO N-D
150 J=I+D
160 IF A(I) =< A(J) THEN 190
```

```
170 U=A(I):A(I)= A(J): A(J)=U
180 TEST=1
190 NEXT
200 IF TEST=1 THEN 130
210 D=INT(D/2)
220 IF D<>0 THEN 130
230 FOR I=1 TO N
240 PRINT A(I);
250 NEXT
```

Exemple

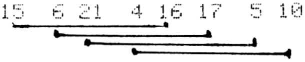
676	351	348	713	540	406	321	15	388	22	127	290	526	409	494
15	351	22	127	290	406	321	494	388	348	713	540	526	409	676
15	351	22	127	290	406	321	494	388	348	713	540	526	409	676
15	290	22	127	351	388	321	494	406	348	409	540	526	713	676
15	290	22	127	351	388	321	409	406	348	494	540	526	713	676
15	290	22	127	351	388	321	409	406	348	494	540	526	713	676
15	22	127	290	351	321	388	406	348	409	494	526	540	676	713
15	22	127	290	321	351	388	348	406	409	494	526	540	676	713
15	22	127	290	321	351	348	388	406	409	494	526	540	676	713
15	22	127	290	321	348	351	388	406	409	494	526	540	676	713

Commentaires

Considérons une liste plus petite (huit éléments) pour simplifier le raisonnement. Soit N=8. On a au départ D=4.

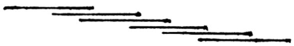
```
-> au début          15  6 21  4 16 17  4 10

-> première boucle   15  6 21  4 16 17  5 10
      D=4
```



```
      D'où           15  6  5  4 16 17 21 10
                         T=1 -> nouveau passage ne
                                conduisant à aucun changement

-> deuxième boucle   15  6 21  4 16 17  5 10
      D=2
```



soit successivement :

```
        soit successivement
 * Premier Passage:  15  4 21  6 16 17  5 10
                     15  4 16  6 21 17  5 10
                     15  4 16  6  5 17 21 10
                     15  4 16  6  5 10 21 17
                         T=1 -> nouveau Passage
 * deuxième Passage: 15  4  5  6 16 10 21 17
                         T=1 -> nouveau Passage ne
                                conduisant à aucun changement
-> troisième boucle
    D=1
 * Premier Passage:   4 15  5  6 16 10 21 17
                      4  5 15  6 16 10 21 17
                      4  5  6 15 16 10 21 17
                      4  5  6 15 10 16 21 17
                      4  5  6 15 10 16 17 21
                         T=1 -> nouveau Passage
 * deuxième Passage:  4  5  6 10 15 16 17 21
                         T=1 -> nouveau Passage ne
                                conduisant à aucun changement
-> quatrième boucle
    D=0        ->   tri terminé.
```

Tri Shell-Metzner

Il débute comme le tri Shell (scindage en deux moitiés de la liste), mais à la fin du premier passage, chaque demi-liste est triée séparément.

Programme

```
10 REM TRI SHELL  METZNER
20 INPUT" NOMBRE N= ";N
30 DIM A(N)
40 FOR I =1 TO N
50 A(I) = INT(RND(1)*1000)
55 PRINT A(I);
60 NEXT
65 PRINT
100 REM  SHELL METZNER
110 D= INT(N/2)
130 J=1: B=N-D
140 C=J
150 E=C+D
160 IF A(C)=<A(E) THEN C=0:GOTO190
```

```
170 U=A(C):A(C)= A(E): A(E)=U
180 C=C-D
190 IF C>=1 THEN 150
200 J=J+1
210 IF J=<B THEN 140
220 D=INT(D/2)
230 IF D<>0 THEN 130
240 FOR I=1 TO N
250 PRINT A(I);
260 NEXT
```

Nous supposons, comme dans les cas précédents, que la matrice à trier est A(15).

```
676  351  348  713  540  406  321  15   388  22   127  290  526  409  494

 15  351  22   127  290  406  321  494  388  348  713  540  526  409  676
 15  290  22   127  351  388  321  409  406  348  494  540  526  713  676
 15  22   127  290  321  348  351  388  406  409  494  526  540  676  713
```

C. Tris par sélection

Sélection

On cherche l'élément le plus grand dans la liste comprenant F éléments et on le range à la tête, F = N, N − 1, ..., 1.

Commentaire

Dans un premier temps, toute la liste est examinée. Le plus grand élément, U, est amené en début de liste. Au passage, l'élément qui était à cet emplacement est amené à l'endroit où a eu lieu la dernière comparaison positive. La liste est ensuite réduite d'un élément et le cheminement du programme reprend sur cette liste raccourcie.

Programme

```
10 REM TRI PAR SELECTION
20 INPUT "NOMBRE N= ";N
30 DIM A(N)
40 FOR I=1 TO N
50 A(I) = INT(RND(1)*1000)
```

```
60 NEXT
100 REM TRI
110 F=N
120 I=1:P=1:U=A(1):IF F=1 THEN 200
130 I=I+1:IF A(I)>U THEN P=I:U=A(I)
140 IF I=F THEN A(P)=A(F):A(F)=U:F=F-1:GOTO 120
150 GOTO 130
200 FOR I=1 TO N
210 PRINT A(I);
220 NEXT
```

Exemple

```
676  351  348  713  540  406  321  15   388  22   127  290  526  409  494

676  351  348  494  540  406  321  15   388  22   127  290  526  409  713
409  351  348  494  540  406  321  15   388  22   127  290  526  676  713
409  351  348  494  526  406  321  15   388  22   127  290  540  676  713
409  351  348  494  290  406  321  15   388  22   127  526  540  676  713
409  351  348  127  290  406  321  15   388  22   494  526  540  676  713
22  351  348  127  290  406  321  15  388  409  494  526  540  676  713
22  351  348  127  290  388  321  15  406  409  494  526  540  676  713
22  351  348  127  290  15  321  388  406  409  494  526  540  676  713
22  321  348  127  290  15  351  388  406  409  494  526  540  676  713
22  321  15  127  290  348  351  388  406  409  494  526  540  676  713
22  290  15  127  321  348  351  388  406  409  494  526  540  676  713
22  127  15  290  321  348  351  388  406  409  494  526  540  676  713
22  15  127  290  321  348  351  388  406  409  494  526  540  676  713
15  22  127  290  321  348  351  388  406  409  494  526  540  676  713
```

Exemple détaillé.
Reprenons la liste utilisée précédemment :

15 6 21 4 16 17 5 10

Au premier passage, 21 est échangé avec 10 :

15 6 10 4 16 17 5 21

Au deuxième passage, 17 est échangé avec 5 :

15 6 10 4 16 5 17 21

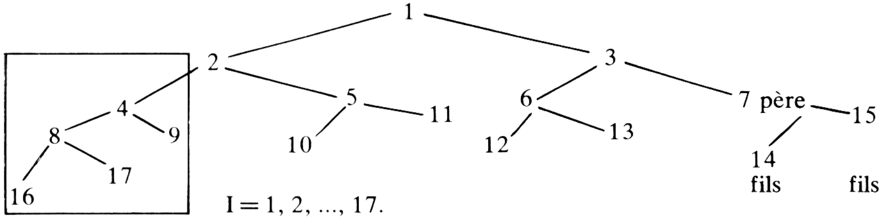
Puis successivement :

15	6	10	4	5	16	17	21
5	6	10	4	15	16	17	21
5	6	4	10	15	16	17	21
5	4	6	10	15	16	17	21
4	5	6	10	15	16	17	21

Heap sort

La méthode est particulièrement bien expliquée dans le livre de D. Knuth : *Sorting and Searching* (pp. 145-149).

Notre programme commenté se fonde sur l'algorithme H (pp. 146-147). L'idée consiste à créer des sous-arbres ordonnés...

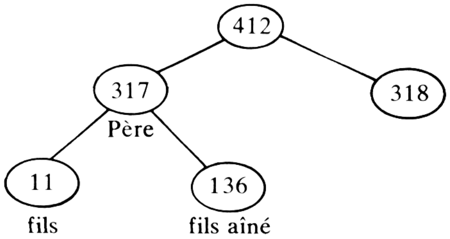


ce qui veut dire :

$$A\left(\text{INT}\left(\frac{I}{2}\right)\right) \geqslant A(I) \text{ si } 1 \leqslant G \leqslant \text{INT}\left(\frac{I}{2}\right) < I \leqslant N\Big)$$

...et à garder cette structure favorable en remontant les grands éléments à la tête.

Sous-arbre rangé



A(4), A(8), A(9), A(16), A(17)

Algorithme H

Soit une liste de N éléments à trier, rangés dans A (1), ..., A (N). A la fin on aura $A(1) \leqslant A(2) \leqslant, ..., \leqslant A(N)$.

Le programme débute ainsi :

- définition de deux pointeurs d'extrémité :

  G pointe sur la gauche de la liste,

  D pointe sur la droite,

  au départ : $G = INT\left(\frac{N}{2}\right) + 1$, $D = N$ (ligne 100)

- la variable A sert de mémoire.

Le tri commence ligne 200 du programme.

Dans un premier temps, on diminue G, et ensuite, arrivé à $G = 1$, D jusqu'à $D = 1$.

A ce moment-là, le tri est terminé si on range l'élément A en bas de liste : $A(1) = A$.

Pendant la première phase ($G > 1$), on crée des sous-arbres prétriés, c'est-à-dire :

$$A\left(INT\left(\frac{I}{2}\right)\right) > A(I) (*)$$

si :

$$G \leqslant INT\left(\frac{I}{2}\right) < I \leqslant N$$

A la fin de ce premier cycle, tous les sous-arbres sont ordonnés et (*) reste valable avec $G = 1$. L'élément le plus grand est évidemment en tête de liste ($= A(1)$).

On peut donc réduire la liste en posant $A = A(D)$, ($D = N$), $A(D) = A(1)$, $D = D - 1$ (ligne 210). Dans la liste restante, qui comprend D éléments, c'est-à-dire un élément de moins, on fait remonter en tête le plus grand élément, qui sera ensuite placé sur A (D), ($D = N - 1$), après avoir posé $A = A(D)$, etc.

A la fin, on arrive à $D=1$ et l'élément le plus petit devient $A(1)$. Comment remonte-t-on les éléments?

En première phase, on a $A=A(G)$ et ensuite $A=A(D)$, dont il faut trouver la bonne place dans le sous-arbre approprié en préservant le préordre.

Dans ce but, on pose d'abord $J=G$ (ligne 300) et ensuite (ligne 400) :

$I=J$ et

$J=2*J$ si $A(2*J) \geqslant A(2*J+1)$ ou

$J=2*J+1$ si $A(2*J) < A(2*J+1)$ (ligne 500)

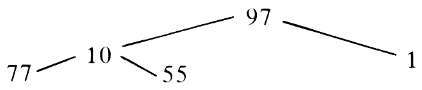
($A(J)$ devient ainsi le *fils aîné* du père $A(I)$.)

Si $A > A(J)$ (ligne 600), on a trouvé la bonne place pour A (l'ordre est respecté) et le programme nous renvoie ligne 200 après avoir posé $A(I)=A$ (ligne 800).
Sinon, on descend par $I=J$, $J=2*J$ ou $J=2*J+1$ etc. et on reprend la boucle.

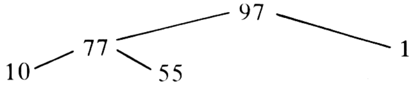
Prenons un exemple ($N=5$) :
Soit la liste de cinq nombres donnée ci-après :

97 10 1 77 55

ou :



On a $G=2$, $D=5$, $A=A(G)=10$ et on échange 10 avec 77 :



$G=1$, rien ne se passe, la première phase est terminée, l'arbre est ordonné.

On réduit la liste en posant :

A = 55, A (5) = 97, D = 4

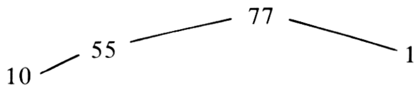
I = 1, J = 2

A (2) $\longrightarrow$ A (1)

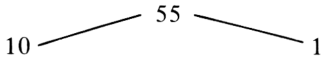
et on reprend :

I = 2, J = 4

On a A > 10 = A (4), donc A $\longrightarrow$ A (2).
Il en résulte :



On pose A = 10, A (4) = 77, D = 3 et 10 reste en bas :



Maintenant :

A = 1, A (3) = 55, D = 2

et finalement :

A (2) = 10 et A (1) = 1

```
10 REM HEAPSORT D'APRES D. KNUTH
20 INPUT "NOMBRE N= ";N
30 DIM A(N)
40 FOR I=1 TO N
50 A(I)= INT(RND( 1)*1000)
60 NEXT
100 G=INT(N/2)+1: D=N
110 REM DEBUT , G GAUCHE , D DROITE
200 IF G>1 THEN G=G-1: A=A(G):GOTO 300
```

```
 210 A=A(D):A(D)=A(1): D=D-1: IF D=1 THE
N A(1)=A:GOTO900
 220 REM DIMINUER G  ET APRES D JUSQU'A
LA FIN D=1
 300 J=G
 310 REM A(INT(K/2))>=A(K), G<INT(K/2)<K
<=D: PREPARATION POUR TRI SOUSARBRES
 400 I=J:J=2*J
 410 IF J<D THEN 500
 420 IF J=D THEN 600
 430 IF J>D THEN 800
 440 REM TRI SOUSARBRE
 500 IF A(J)<A(J+1) THEN J=J+1
 510 REM L'AUTRE FILS DU PERE A(I) PLUS
GRAND?
 600 IF A>= A(J) THEN 800
 610 REM COMPARAISON PLACE TROUVEE?
 700 A(I)=A(J):GOTO 400
 710 REM SOUSTRI PAS FINI, ON REMONTE
 800 A(I)=A: GOTO 200
 810 REM SOUSTRI TERMINE
 900 FOR I=1 TO N
 910 PRINT A(I);
 920 NEXT
```

LISTE
G D A

```
676  351  348  713  540  406  321  15  388  22  127  290  526  409  494
  7  15  321
676  351  348  713  540  406  494  15  388  22  127  290  526  409  321
  6  15  406
676  351  348  713  540  526  494  15  388  22  127  290  406  409  321
  5  15  540
676  351  348  713  540  526  494  15  388  22  127  290  406  409  321
  4  15  713
676  351  348  713  540  526  494  15  388  22  127  290  406  409  321
  3  15  348
676  351  526  713  540  406  494  15  388  22  127  290  348  409  321
  2  15  351
676  713  526  388  540  406  494  15  351  22  127  290  348  409  321
  1  15  676
713  676  526  388  540  406  494  15  351  22  127  290  348  409  321
  1  14  321
676  540  526  388  321  406  494  15  351  22  127  290  348  409  713
  1  13  409
540  409  526  388  321  406  494  15  351  22  127  290  348  676  713
  1  12  348
526  409  494  388  321  406  348  15  351  22  127  290  540  676  713
  1  11  290
494  409  406  388  321  290  348  15  351  22  127  526  540  676  713
  1  10  127
409  388  406  351  321  290  348  15  127  22  494  526  540  676  713
  1  9  22
406  388  348  351  321  290  22  15  127  409  494  526  540  676  713
  1  8  127
388  351  348  127  321  290  22  15  406  409  494  526  540  676  713
  1  7  15
351  321  348  127  15  290  22  388  406  409  494  526  540  676  713
  1  6  22
348  321  290  127  15  22  351  388  406  409  494  526  540  676  713
  1  5  22
321  127  290  22  15  348  351  388  406  409  494  526  540  676  713
  1  4  15
290  127  15  22  321  348  351  388  406  409  494  526  540  676  713
  1  3  22
127  22  15  290  321  348  351  388  406  409  494  526  540  676  713
  1  2  15
22  15  127  290  321  348  351  388  406  409  494  526  540  676  713
  1  1  15
15  22  127  290  321  348  351  388  406  409  494  526  540  676  713
```

LISTE PRECEDENTE N=5
G D A

```
97  10  1  77  55
  2  5  10
97  77  1  10  55
  1  5  97
97  77  1  10  55
  1  4  55
77  55  1  10  97
  1  3  10
55  10  1  77  97
  1  2  1
10  1  55  77  97
  1  1  1
1  10  55  77  97
```

Exemples, tableau comparatif

```
   Tableau comparatif des différentes méthodes de tri
    ( Microordinateur familial, équipé d'un 6502 à 1MHz )

  Analyse: N  nombre d'éléments
```

Tri	Durée du tri	Liste N=200	Liste N=100 Prétriée
Ripple	O(N^2)	8 min	95 sec
Shaker	O(N^2)	7.5 min	2 min
Bubble	O(N^2)	7 min	110 sec
Quick LP=4 - Ripple	O(NlogN)	1 min	25 sec
Quick LP=9-Insertion	O(NlogN)	45 sec	18 sec
Insertion	O(N^2)	4 min 45 sec	85 sec
Shell	O(N^a),1<a<=1.5	2 min 30 sec	45 sec
Shell Metzner	O(N^b),1<b<=1.5	1 min 25 sec	35 sec
Selection	O(N^2)	6 min 30 sec	100 sec
Heapsort	O(NlogN)	80 sec	35 sec

Les tris Quick, Shell-Metzner et Heap sort sont particulièrement efficaces et recommandables.

18. ACCÈS A UN ÉLÉMENT DE FICHIER

Le traitement de fichiers s'effectue souvent à l'aide d'un lecteur de disquettes, associé à son DOS *(Disk Operating System)*. Celui-ci permet en général de créer des fichiers à accès séquentiel (les *rubriques* sont disposées les unes à la suite des autres) et des fichiers à accès direct (à chaque rubrique est associé un numéro d'enregistrement). Nous entendons par *rubrique* un ensemble de données ayant un point commun. Pour un annuaire, par exemple, une rubrique contient : nom, prénom, adresse, numéro de téléphone. Si le fichier est important, la recherche d'une rubrique peut se faire :

- séquentiellement, pour un fichier à accès séquentiel : il faut lire dans l'ordre toutes les fiches jusqu'au moment où l'on trouve celle que l'on cherche. Une telle méthode est très pénalisante en temps si le fichier est très important;

- directement, pour un fichier à accès direct, si l'on connaît le numéro de la rubrique cherchée, ce qui n'est pas le cas en général, car à priori rien n'associe le nom « DUPONT » à la rubrique n° 15!

Nous proposons deux méthodes permettant donc d'aboutir à un tel résultat.

ACCÈS PAR CLÉ

On constitue en mémoire centrale une table d'index. Celle-ci est une matrice de chaînes; l'indice de chaque variable est justement le numéro de rubrique du fichier à accès direct sur disquette.

A$(1) = « DUPONT »
A$(2) = « DURAND »
A$(3) = « DURANT »
.
.
.
A$(N) = « ZORGON »

Pour un fichier figé, une telle matrice est définitive. La recherche du numéro de rubrique est alors aisée :

```
10 INPUT"NOM CHERCHE=?";A$
20 FOR U=1 TO N
30 IF A$(U)=A$ THEN M=U
40 NEXT U
50 PRINT"LE NUMERO CHERCHE EST : ";M
```

Il suffit maintenant d'aller chercher sur la disquette la rubrique M pour pouvoir l'exploiter.

Le traitement d'un fichier dynamique est plus délicat; il faut pouvoir faire des suppressions et des rajouts.

- Pour un ajout, il faut transférer le fichier en mémoire centrale, le compléter avec la nouvelle rubrique puis le trier, on peut alors créer la nouvelle table d'index.
- Pour un retrait, l'opération est plus simple, puisqu'il suffit alors de décaler *vers le haut* l'ensemble des rubriques suivant celle qui a été supprimée. La table d'index est alors construite.

HASH-CODING

La difficulté, comme nous l'avons vu précédemment, est en général d'associer le numéro de rubrique à la *clé*, c'est-à-dire la chaîne de caractères correspondante.

On peut donc associer à chaque clé un numéro (qui sera celui de la rubrique concernée), *déduit* des propriétés de la clé elle-même. On pourrait ainsi faire correspondre à chaque mot un nombre qui serait, par exemple, la somme des positions de chacune de ses lettres dans l'alphabet :

D ⟶	4	D	4
U ⟶	21	U	21
R ⟶	18	P	16
A ⟶	1	O	15
N ⟶	14	N	14
D ⟶	4	T	20
	62		90

Rubrique 90 : DUPONT Jacques,...

Rubrique 62 : DURAND Paul,...

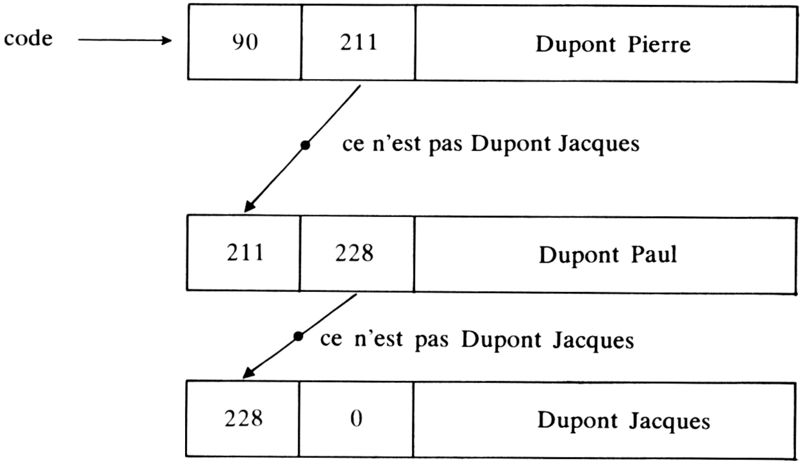
Deux problèmes surgissent alors :

- Il faut prévoir que le nombre N trouvé dans ces conditions puisse être compris entre 1 et le nombre maximal K de rubriques du fichier; on est d'ailleurs obligé de surdimensionner celui-ci, puisqu'il n'y a aucune chance que, à K mots, correspondent K nombres tous différents et compris entre 1 et K!
- En outre, et c'est une conséquence, il n'est pas impossible (il est même très probable si le fichier est important) qu'à deux clés différentes corresponde le même code. On dit alors qu'il y a *collision*. Ce risque est d'autant plus réduit que le nombre de rubriques du fichier est plus grand que le nombre réel de rubriques. Cependant, une telle éventualité n'est jamais à écarter.

La méthode de gestion des collisions peut alors être la suivante : on associe à chaque clé, dans le fichier, un numéro. Celui-ci sera égal à zéro s'il n'y a pas de collision, ou bien au numéro de rubrique de la clé portant le même code.

Exemple

On recherche Dupont Jacques, sachant que le fichier contient en sus Dupont Paul et Dupont Pierre. Le code n'est calculé que sur le nom, soit par exemple 90.



Ainsi, pour un fichier de 200 rubriques, on peut affecter 250 places, les indices 201 à 250 correspondant alors au fichier des collisions. On se reportera à la référence citée précédemment pour plus de détails.

INDEX

Cet ouvrage propose une collection d'algorithmes numériques, destinés à être employés sur micro-ordinateur dans le but de résoudre nombre de problèmes scientifiques (mathématiques, physique, électronique). Les programmes de ce livre sont écrits en BASIC Microsoft et ne font appel à aucune propriété particulière d'une machine.

LES AUTEURS

Pierre Beaufils est professeur agrégé de physique et Wolfram Luther est docteur en mathématiques. Ils sont auteurs de plusieurs ouvrages traitant de l'utilisation des micro-ordinateurs dans l'enseignement secondaire.

0359 1287 198 F

9 782736 103590

SYBEX

BASIC APPROFONDI ALGORITHMES MATHÉMATIQUES ET PHYSIQUES



